



# RVS TECHNICAL CAMPUS - COIMBATORE

(An Autonomous Institution)

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai.

Recognized By UGC and Accredited by NAAC with 'A' Grade.

Kumaran Kottam Campus, Kannampalayam, Coimbatore - 641 402.



## Sales Automobile Using Salesforce CRM

### Project Team Members

1. Ishwarya K (56A7EE8416A99E9AEEC43F21CB0951BD)
2. Keerthiga I (62E80265908CA99B17442830CE5A3FD8)
3. Ajitha S (0B718EB0EFC1ADB829240CEC7F67B555)
4. Vishnukumar S (D40675BFE16B607AAB84F02E8A607289)

# Sales Automobile Using Salesforce CRM

## 1. Project Overview

This project is focused on [Project Name/Objective], designed to address [the primary challenge or opportunity]. The goal is to deliver a comprehensive solution by leveraging [specific technology, platform, or approach]. Through this project, we aim to enhance [key benefit: e.g., operational efficiency, user experience, data accuracy] and support the long-term goals of [organization/department/business].

## 2. Objectives

- **Optimize Lead Management:** Use Salesforce's lead management tools to capture, nurture, and track potential customers.
- **Automate Sales Process:** Implement automation features like workflow rules, approvals, and process builders to speed up and simplify the sales cycle.
- **Enhance Customer Engagement:** Use Salesforce's communication tools (email, chat, etc.) to maintain continuous contact with customers, providing updates and personalized information.
- **Data-Driven Decision Making:** Utilize Salesforce's reporting and analytics features to track key performance indicators (KPIs), sales trends, and customer behavior for strategic decision-making.
- **Improve After-Sales Service:** Leverage Salesforce Service Cloud to provide post-sale support and build long-term customer relationships.

## 3. Salesforce Key Features and Concepts Utilized

### a. *Lead Generation & Management*

- **Lead Capture:** Integrate web forms and marketing tools (like email campaigns or social media ads) to capture leads and feed them into Salesforce.

- **Lead Scoring:** Use custom scoring criteria to prioritize leads based on likelihood to convert, helping sales teams focus on high-value opportunities.
- **Lead Nurturing:** Automate follow-up emails and reminders for sales representatives to engage with potential customers at the right time.

#### *b. Sales Pipeline & Opportunity Management*

- **Sales Process Customization:** Tailor Salesforce to match the specific stages of the automobile sales process (e.g., Inquiry > Test Drive > Negotiation > Closing).
- **Opportunity Management:** Track the progress of each deal, from initial inquiry to closing, with detailed information on customer preferences, trade-in details, financing options, etc.
- **Sales Forecasting:** Utilize Salesforce's forecasting tools to predict future sales and allocate resources effectively.

#### *c. Customer Relationship & Engagement*

- **Customer Profiles:** Maintain detailed customer profiles, including contact information, past purchases, service history, and preferences.
- **Personalized Communication:** Send personalized emails, SMS, or notifications about special offers, new arrivals, or service reminders based on customer data.
- **360-Degree View:** Integrate sales, service, and marketing data to provide a unified view of each customer's interactions with the business.

#### *d. Reporting & Analytics*

- **Dashboards:** Create customized dashboards to track key metrics such as sales performance, lead conversion rates, and customer satisfaction.
- **Reports:** Generate detailed reports for management to monitor team performance, sales trends, and other KPIs to drive business decisions.
- **AI-Powered Insights:** Leverage Salesforce's AI-powered tools (like Einstein Analytics) to gain predictive insights into customer behaviors and market trends.

### e. Integration with Other Systems

- **ERP Systems:** Integrate Salesforce with the company's ERP system to provide seamless access to inventory, pricing, and finance data.
- **Third-Party Tools:** Connect Salesforce with other marketing automation tools (like Mailchimp or Marketo) to streamline lead nurturing and engagement..

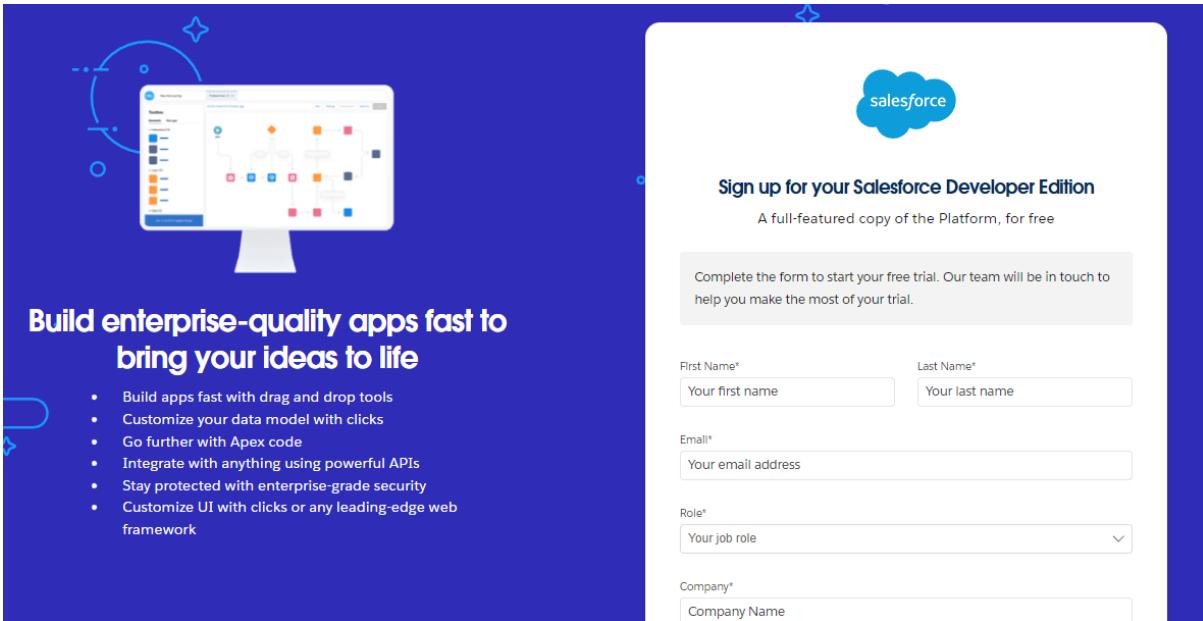
## 4. Detailed Steps to Solution Design

### Step :1

#### 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 :



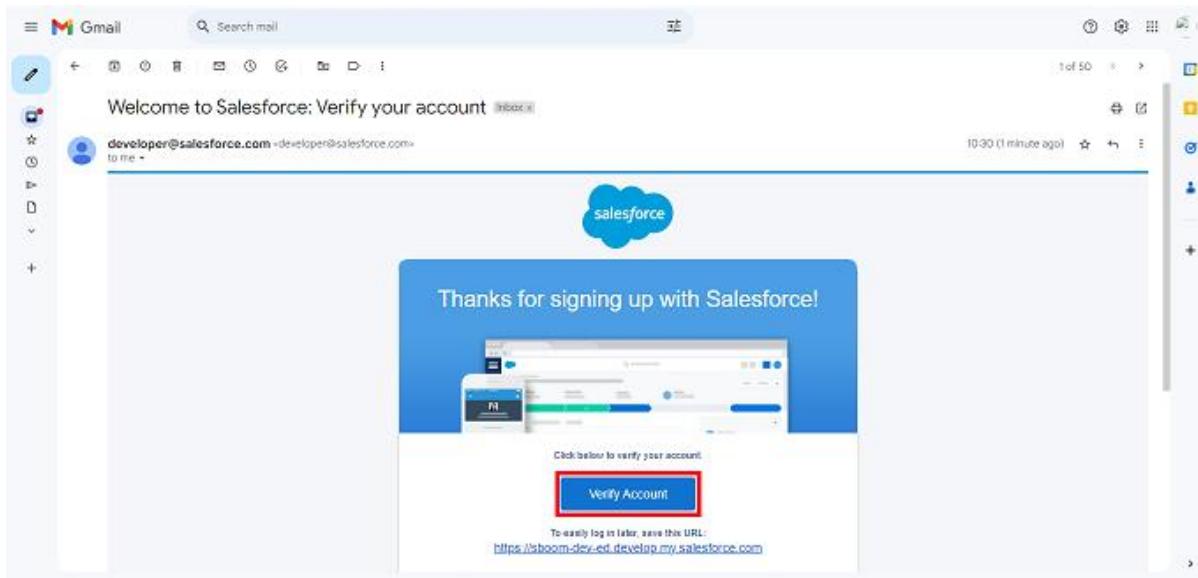
1. First name & Last name
2. Email
3. Role : Developer
4. Company : College Name
5. County : India
6. Postal Code : pin code
7. Username : should be a combination of your name and company

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.

## Step :2

### 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 **lead@sb.oom**.

Make sure to include at least:

- 8 characters
- 1 letter
- 1 number

\* New Password

 Good

\* Confirm New Password

 Match

Security Question

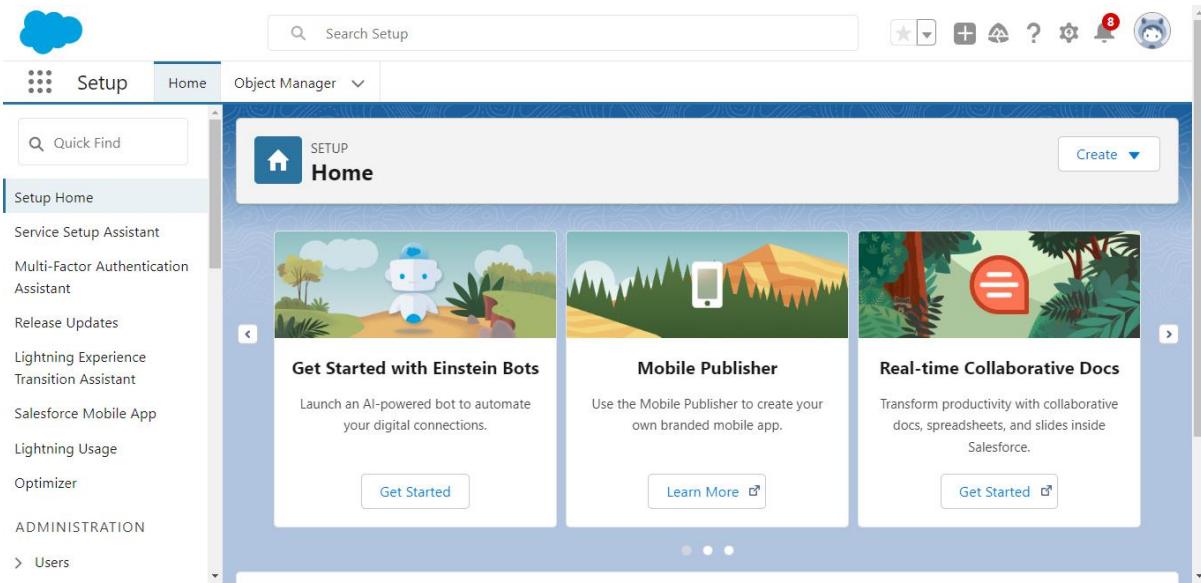
In what city were you born?

\* Answer

asdfghjkl

**Change Password**

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



The screenshot shows the Salesforce Setup Home page. At the top, there's a navigation bar with a cloud icon, the word "Setup", and "Home". Below the navigation is a search bar labeled "Search Setup" and various icons for configuration. On the left, a sidebar lists several setup tools: Service Setup Assistant, Multi-Factor Authentication Assistant, Release Updates, Lightning Experience Transition Assistant, Salesforce Mobile App, Lightning Usage, Optimizer, and Administration. Under Administration, there's a link to "Users". The main content area is titled "SETUP Home" and features three cards: "Get Started with Einstein Bots", "Mobile Publisher", and "Real-time Collaborative Docs". Each card has a brief description and a "Get Started" or "Learn More" button.

## Step :3

### 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.

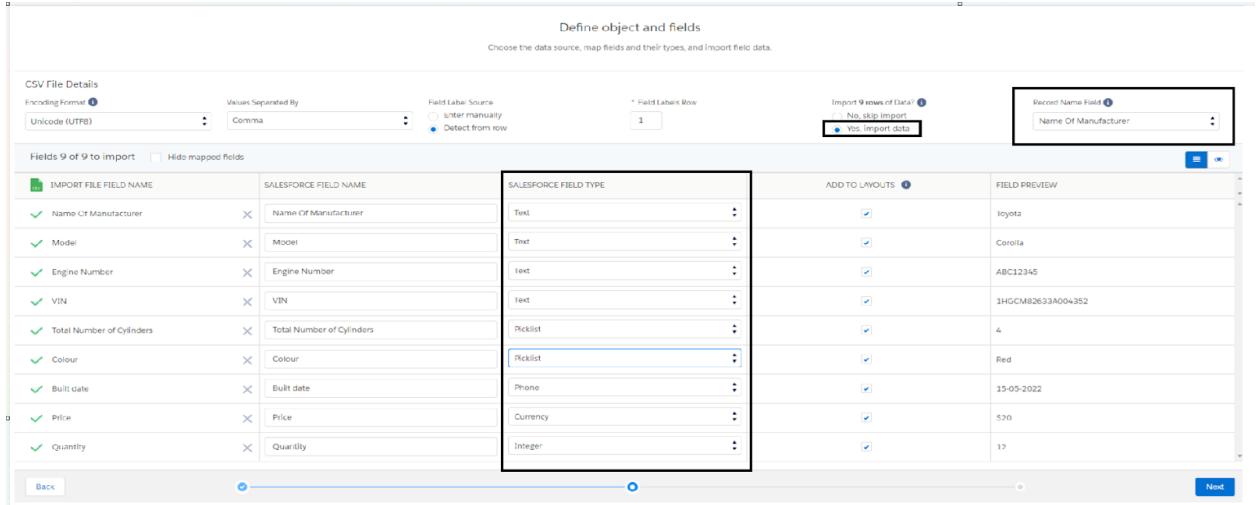


4. Click Create.
5. Select Custom Object from Spreadsheet.



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"/>	Ioyota
✓ Model	Model	Text	<input checked="" type="checkbox"/>	Corolla
✓ Engine Number	Engine Number	Text	<input checked="" type="checkbox"/>	ABC12345
✓ VIN	VIN	Text	<input checked="" type="checkbox"/>	1HGCMB2633A04252
✓ 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	Phone	<input checked="" type="checkbox"/>	15-05-2022
✓ Price	Price	Currency	<input checked="" type="checkbox"/>	520
✓ Quantity	Quantity	Integer	<input checked="" type="checkbox"/>	12

12. Click Next and enter the following settings.
13. Click Finish. The Automobile Information object is successfully created and data imported, all within minutes.

## Step :4

### 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.

**Define object and fields**

Choose the data source, map fields and their types, and import field data.

CSV File Details	Values Separated By	Field Label Source	* Field Labels Row	Import 0 rows of Data	Record Name Field
Encoding Format: Unicode (UTF8)	Comma	<input type="radio"/> Enter manually <input checked="" type="radio"/> Detect from row	1	<input checked="" type="radio"/> No, skip import <input type="radio"/> Yes, import data	Invoice ID
<b>Fields 5 of 5 to import</b> <input type="checkbox"/> Hide mapped fields					
 IMPORT FILE FIELD NAME	<b>SALESFORCE FIELD NAME</b>	<b>SALESFORCE FIELD TYPE</b>	<b>ADD TO LAYOUTS</b>	<b>FIELD PREVIEW</b>	
✓ Invoice ID	Invoice ID	Text	<input checked="" type="checkbox"/>		
✓ Total Price	Total Price	Integer	<input checked="" type="checkbox"/>		
✓ Quantity	Quantity	Integer	<input checked="" type="checkbox"/>		
✓ Unit Price	Unit Price	Integer	<input checked="" type="checkbox"/>		
✓ Purchase Date	Purchase Date	Date	<input checked="" type="checkbox"/>		

## Step :5

### Create Automobile Object

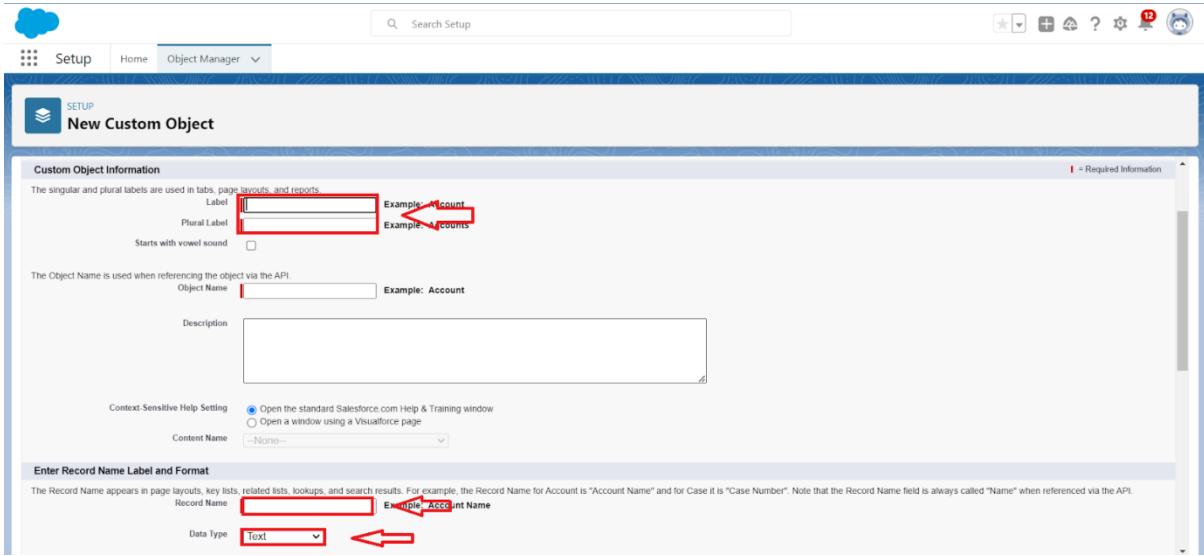
The purpose of creating an Automobile custom object is to store and manage information about Invoice.

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



**SETUP**

## New Custom Object

**Custom Object Information**

The singular and plural labels are used in tabs, page layouts, and reports.

Label	<input type="text" value="Account"/> Example: Account
Plural Label	<input type="text" value="Accounts"/> 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="Account"/> 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

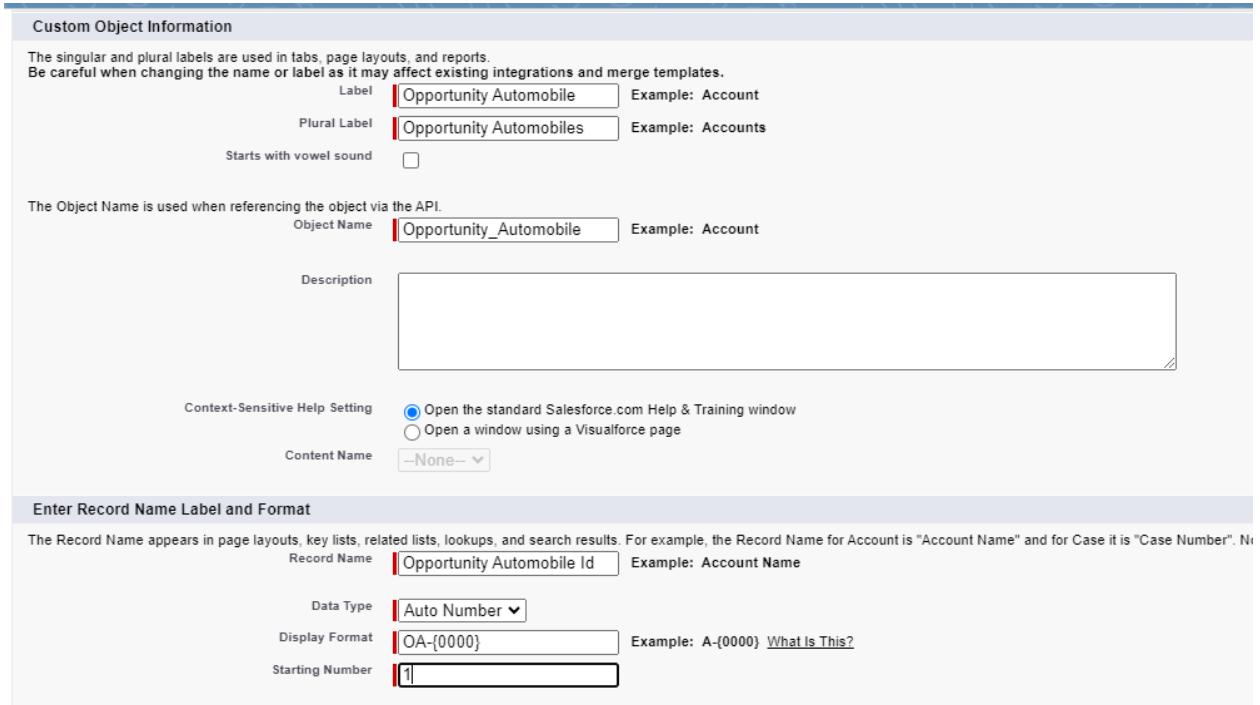
Enter Record Name Label and Format

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". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name	<input type="text" value="Account"/> Example: Account
Data Type	<input type="text" value="Text"/> 

### 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

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". Note that the Record Name field is always called "Name" when referenced via the API.

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

### 2. Click on Allow reports.

3. Allow search

4. Save.

## Step :6

### 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

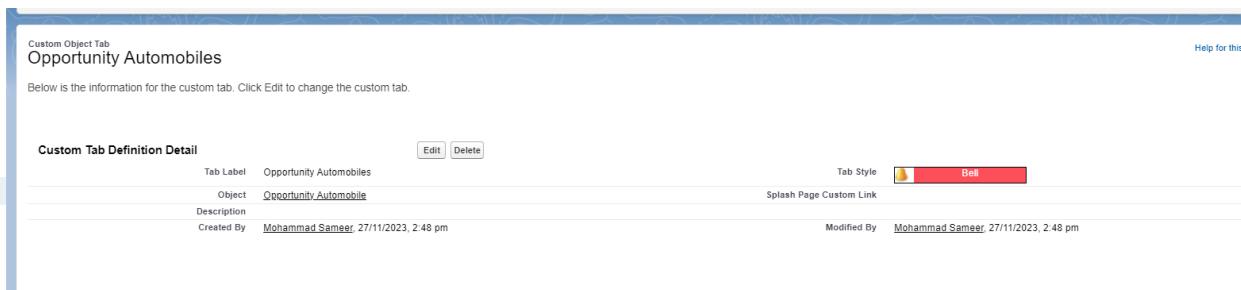
You can create new custom tabs to extend Salesforce functionality or to build new application functionality.

Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external content, allowing you to embed Visualforce 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.



Section	Tab Style	Action Buttons
Custom Object Tabs	Standard	New   What Is This?
Web Tabs	Standard	New   What Is This?

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.



Custom Tab Definition Detail		Edit   Delete	
Tab Label	Opportunity Automobiles	Tab Style	Bell
Object	Opportunity Automobile	Splash Page	Custom Link
Description		Created By	Mohammad Sameer, 27/11/2023, 2:48 pm
Created By	Mohammad Sameer, 27/11/2023, 2:48 pm	Modified By	Mohammad Sameer, 27/11/2023, 2:48 pm

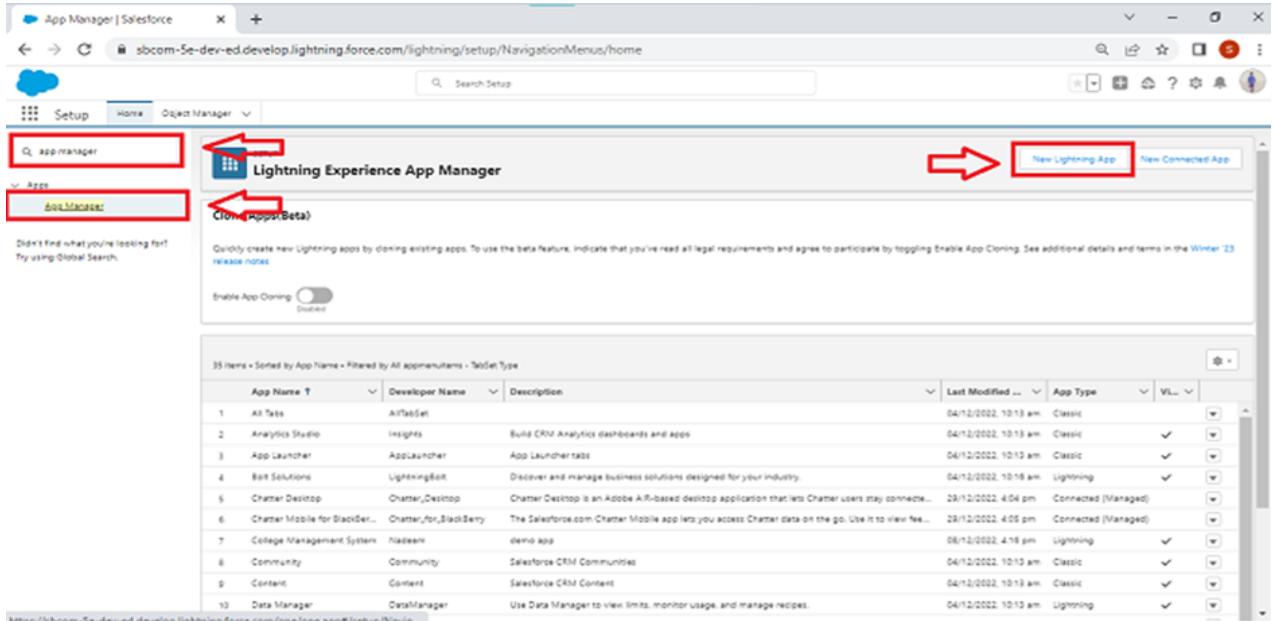
Note: Tabs for Automobile Information & Invoice objects do get created automatically. We do not

need to create tabs for those objects.

## Step :7

### Create a Lightning App

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



The screenshot shows the Salesforce App Manager interface. At the top, there's a search bar with 'Search Setup' and a 'New Lightning App' button. Below the search bar, there are several navigation links: 'Setup', 'Home', 'Object Manager', 'App Manager' (which is highlighted with a red box), and 'App Manager(beta)' (also highlighted with a red box). A large red arrow points to the 'New Lightning App' button. The main area displays a table of existing apps, with the first few rows listed below:

App Name	Developer Name	Description	Last Modified	Type
All Tabs	ArifSiddiq	Build CRM Analytics dashboards and apps	04/12/2022, 10:13 am	Classic
Analytics Studio	Insights	Build CRM Analytics dashboards and apps	04/12/2022, 10:13 am	Classic
App Launcher	AppLauncher	App Launcher tabs	04/12/2022, 10:13 am	Classic
Bolt Solutions	LightningBolt	Discover and manage business solutions designed for your industry	04/12/2022, 10:18 am	Lightning
Chatter Desktop	Chatter/Desktop	Chatter Desktop is an Adobe AIR-based desktop application that lets Chatter users stay connected...	29/12/2022, 4:04 pm	Connected (Managed)
Chatter Mobile for BlackBerry	ChatterForBlackBerry	The Salesforce.com Chatter Mobile app lets you access Chatter data on the go. Use it to view fe...	29/12/2022, 4:05 pm	Connected (Managed)
College Management System	Naresh	demo app	08/12/2022, 4:18 pm	Lightning
Community	Community	Salesforce CRM Communities	04/12/2022, 10:13 am	Classic
Content	Content	Salesforce CRM Content	04/12/2022, 10:13 am	Classic
Data Manager	DataManager	Use Data Manager to view limits, monitor usage, and manage recipes.	04/12/2022, 10:13 am	Lightning

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

Then click Next >> (App option page) keep it as default >> Next >> (Utility Items) keep it as default >> Next.

New Lightning App

### App Details & Branding

Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.

**App Details**

**App Name:**  (Required) Name your app..

**Developer Name:**

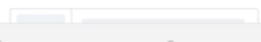
**Description:**

**App Branding**

**Image:**  #0070D2

**Primary Color Hex Value:**

**Org Theme Options:**  Use the app's image and color instead of the org's custom theme

**App Launcher Preview:** 

Next

**Add**
**Navigation**
**Items:**

## New Lightning App

### Navigation Items

They 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**

  X

 Create ▾

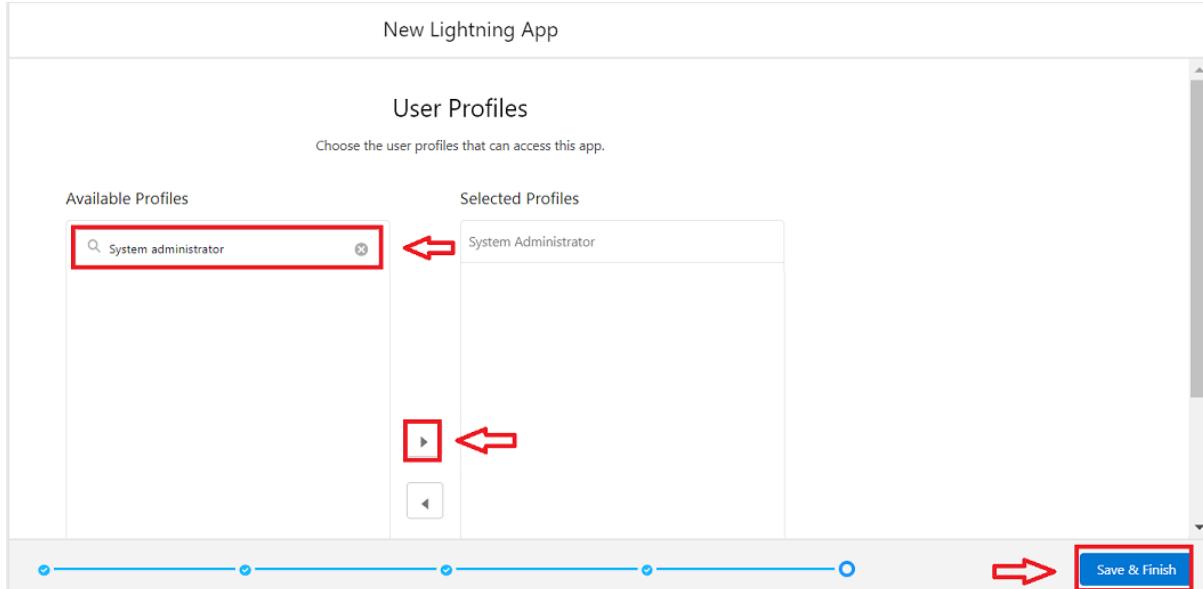
 

**Selected Items**

 Accounts
 Contacts
 Opportunities
 Automobile Information
 Automobiles
 Invoice
 Reports
 Dashboards

▲ ▼

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.



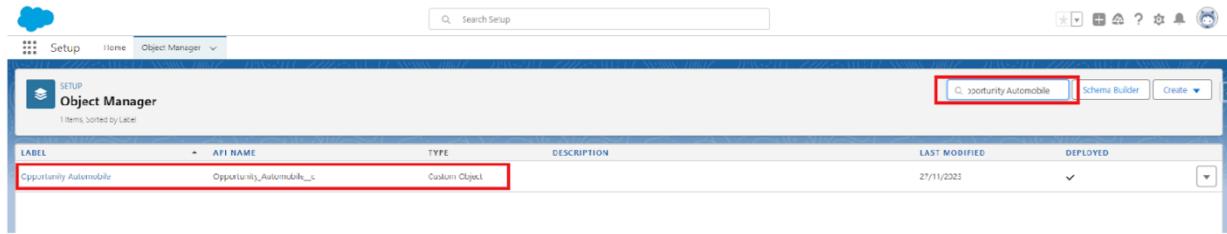
6. Search profiles (System administrator) in the search bar >> click on the arrow button >> save & finish.

## Step :8

### Creating Opportunity Master Detail Relationship Field in Opportunity Automobile Object

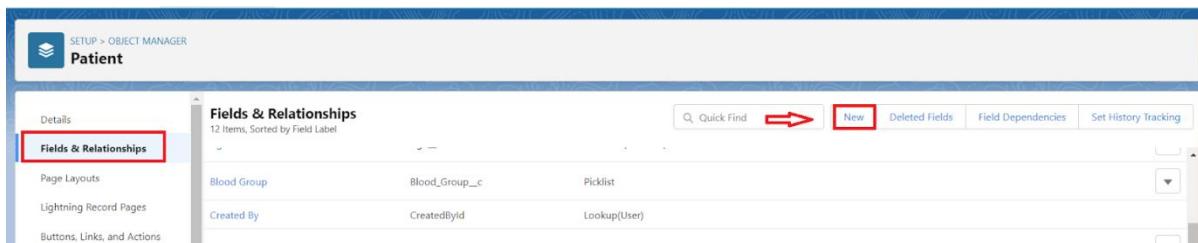
To create fields in an object:

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



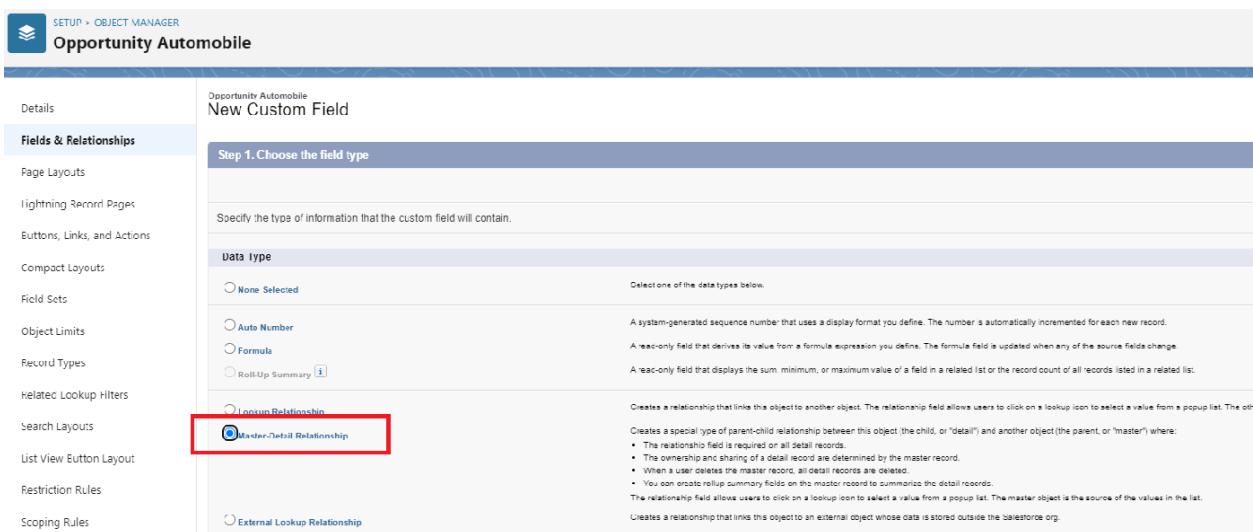
The screenshot shows the Salesforce Object Manager interface. A red box highlights the 'Opportunity Automobile' object in the list, which is a custom object named 'Opportunity\_Automobile\_c'. The interface includes standard Salesforce navigation elements like Setup, Home, and Object Manager.

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



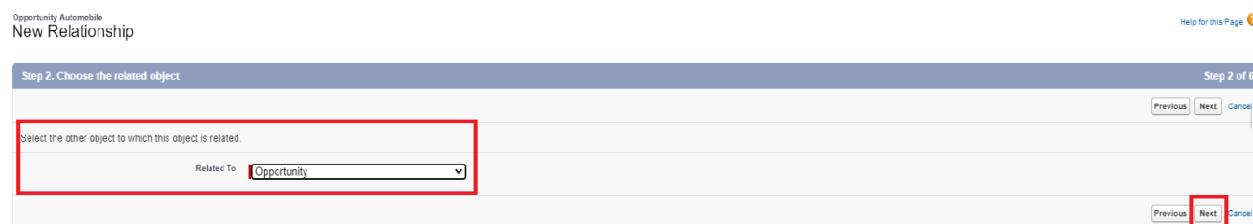
The screenshot shows the 'Fields & Relationships' section for the 'Patient' object. A red box highlights the 'New' button. The list shows fields like 'Blood Group' and 'Created By'.

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



The screenshot shows the 'New Custom Field' setup for the 'Opportunity Automobile' object. The 'Data type' section is open, and a red box highlights the 'Master-Detail Relationship' option. Other options shown include None Selected, Auto Number, Formula, Roll-Up Summary, and External Lookup Relationship.

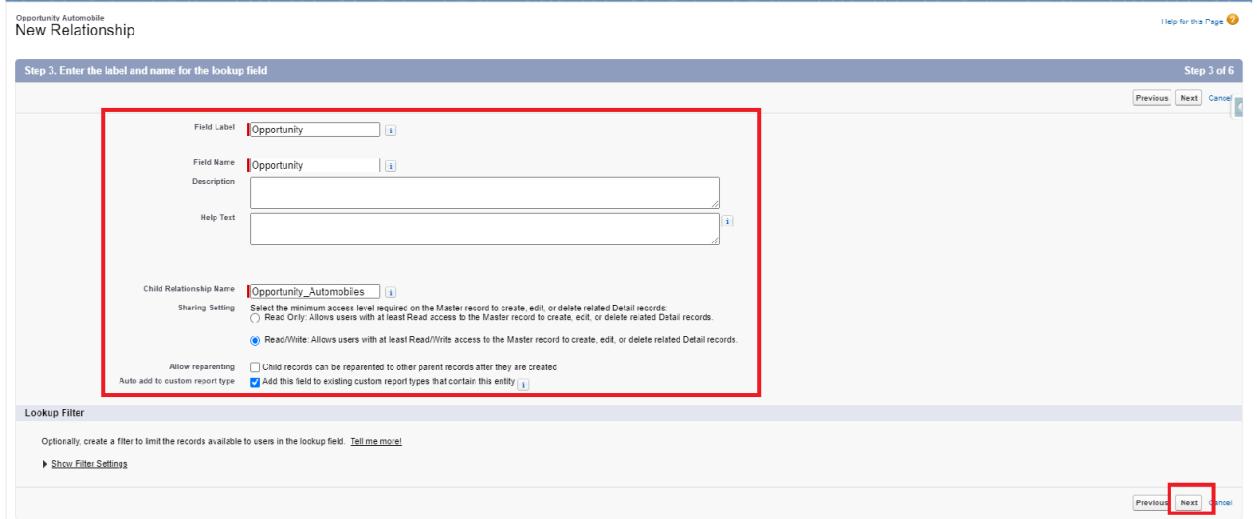
4. Click on Next



The screenshot shows 'Step 2: Choose the related object' of the custom field setup. A red box highlights the 'Related To' dropdown menu, which is set to 'Opportunity'. The bottom right corner shows the 'Next' button highlighted.

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.

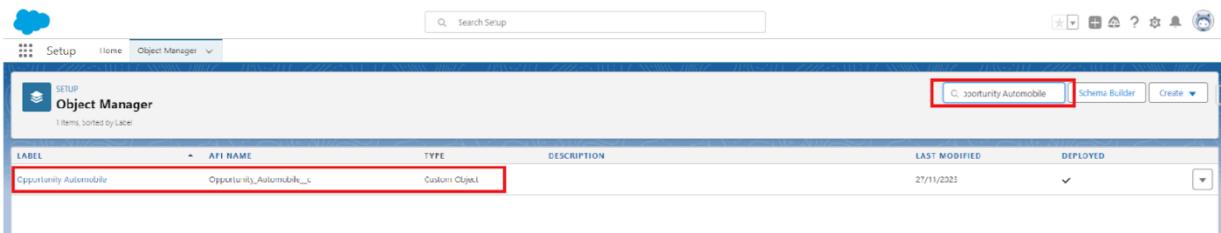


## Step :9

### 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:

- Field Label: Automobile
- Field Name : Automobile

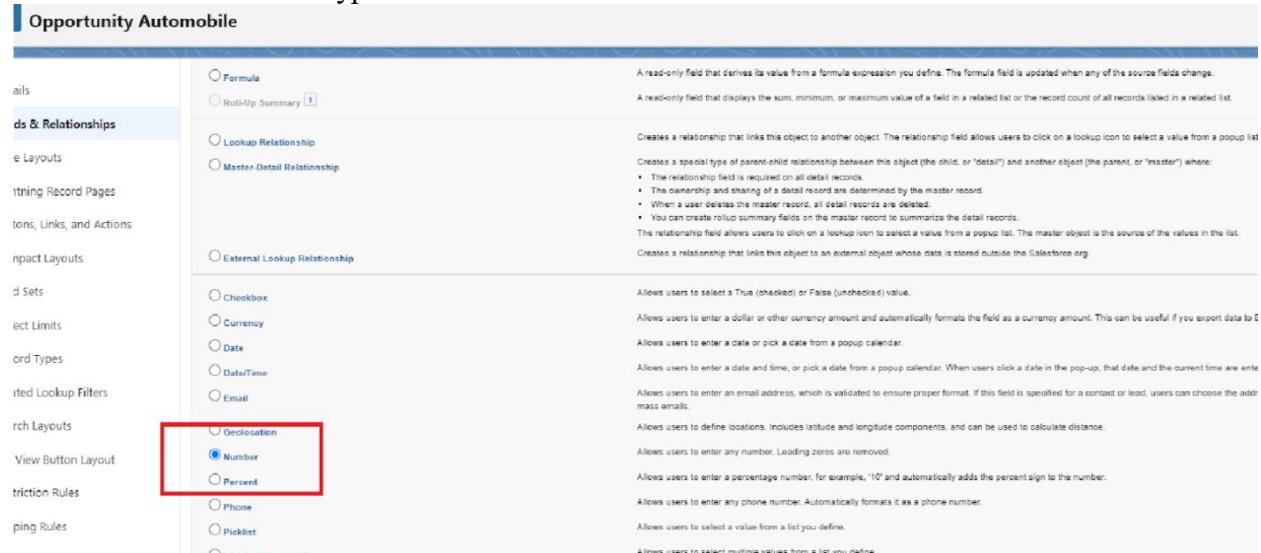
Click on Next >> Next>> Save and new

## Step :10

Creating Quantity Number Field in Opportunity Automobile Object

To create fields in an object:

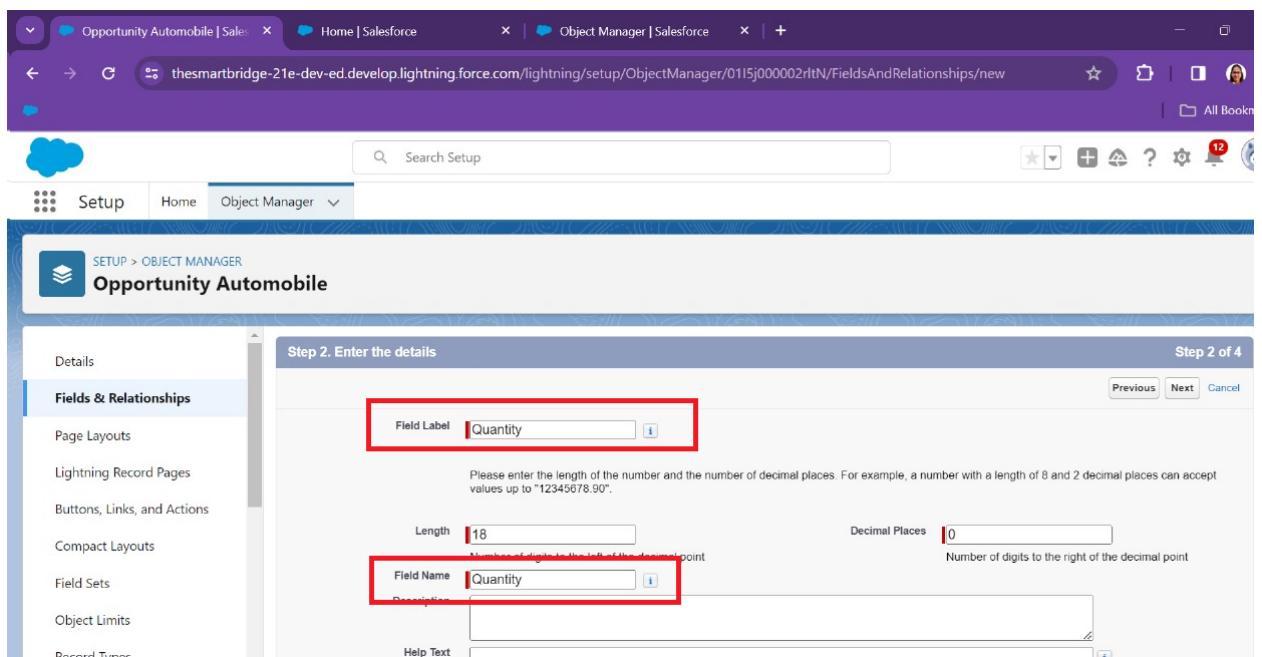
1. Go to setup >> click on Object Manager >> type object name(Opportunity Automobile) in quick find bar >> click on the object.
2. Now click on “Fields & Relationships” >> New.
3. Select Data type as “Number” and click Next.



The screenshot shows the 'Fields & Relationships' section of the Object Manager for the 'Opportunity Automobile' object. On the left, there's a sidebar with various configuration options like Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, and Record Types. The main area lists different field types: Formula, Rollup Summary, Lookup Relationship, Master-Detail Relationship, External Lookup Relationship, Checkbox, Currency, Date, Date/Time, Email, Geolocation, Number (which is selected and highlighted with a red box), Percent, Phone, Picklist, and Picklist (Multi-Select). Each field type has a brief description and some bullet points below it.

a. Field Label >> Quantity

b. Field Name >> Quantity



The screenshot shows the 'Step 2, Enter the details' screen for creating a new field. The left sidebar shows 'Details' and 'Fields & Relationships'. The main form has 'Field Label' set to 'Quantity' (highlighted with a red box) and 'Field Name' set to 'Quantity' (also highlighted with a red box). Below these fields are 'Length' (set to 18) and 'Decimal Places' (set to 0). There's also a 'Help Text' field and a note about number format. At the bottom right, there are 'Previous', 'Next', and 'Cancel' buttons.

Check that Required Check box.

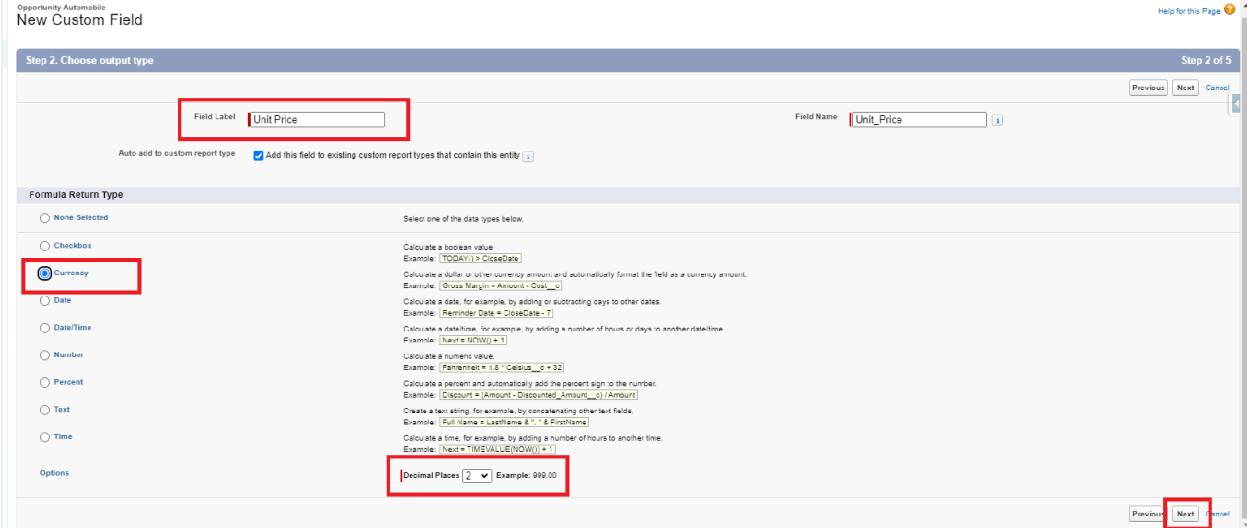
Click Next >> Next >> Save & New.

## Step :11

### 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 Step 2 of 5

Field Label  Field Name

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

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: `{TODAY() > CloseDate}`

Calculate a dollar or other currency amount and automatically format the field as a currency amount.  
Example: `(Gross Margin * Amount) - Cost_Lic`

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

Calculate a datetime, for example, by adding a number of hours or days to another datetime.  
Example: `Next 2 M(W) + 1`

Calculate a numeric value  
Example: `Parentment * 1 * Celsius__c + 32`

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

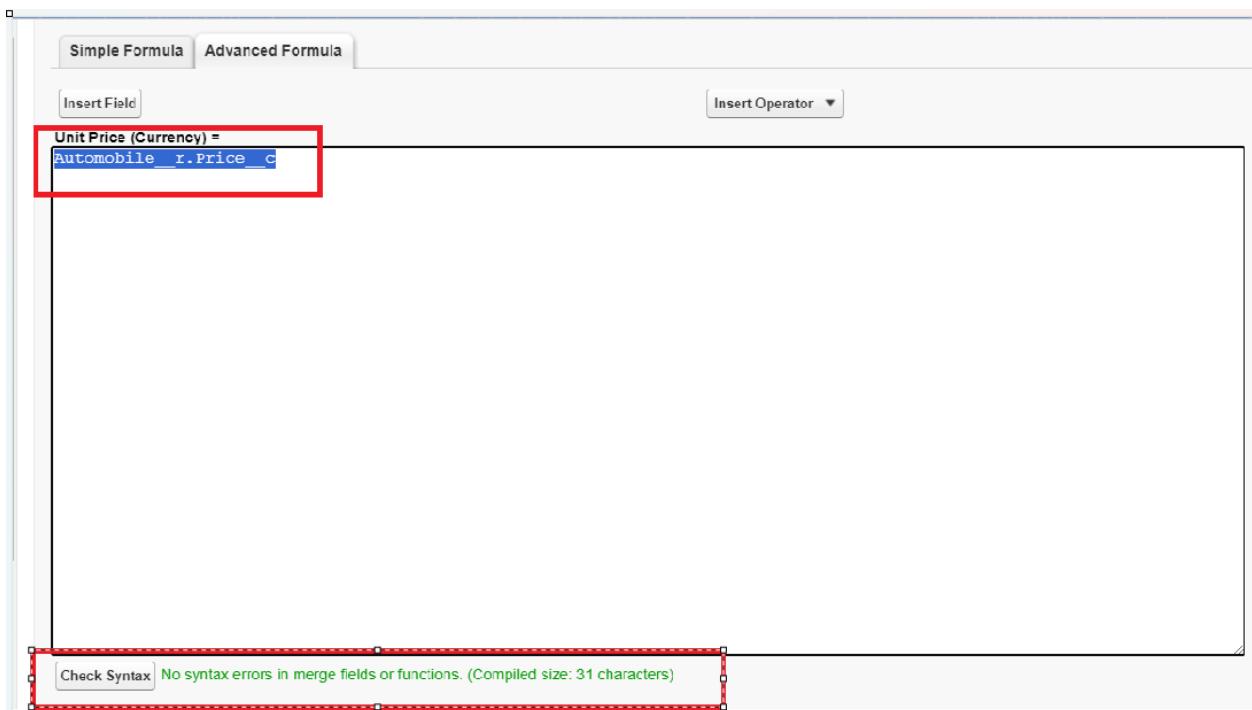
Create a text string, for example, by concatenating other text fields.  
Example: `Full Name = LastName + ", " & FirstName`

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

Decimal Places  Example: 999.00

Previous  Next Cancel

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



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

## Step :12

### 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.

Opportunity Automoblie  
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 entry [\[?\]](#)

Formula Return Type

None Selected

Currency  

Date

DateTime

Number

Percent

Text

Time

Options

Selected one of the datatypes below.

Calculate a boolean value  
Example: `[!Opportunity.CloseDate > Today]`

Calculate a dollar or other currency amount and automatically format the field as a currency amount.  
Example: `[Gross Margin - Amount - Cost__c]`

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

Calculate a datetime, for example, by adding a number of hours or days to another datetime.  
Example: `[Text + 1000000000000000000]`

Calculate a numeric value.  
Example: `[Rate * 1.0 - Gestus__c * 32]`

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 + TIMEVALUE(NOW()) + 1]`

Previous   Next   Cancel

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

Example:  [More Examples...](#)

Simple Formula  Advanced Formula

No syntax errors in merge fields or functions. (Compiled size: 75 characters)

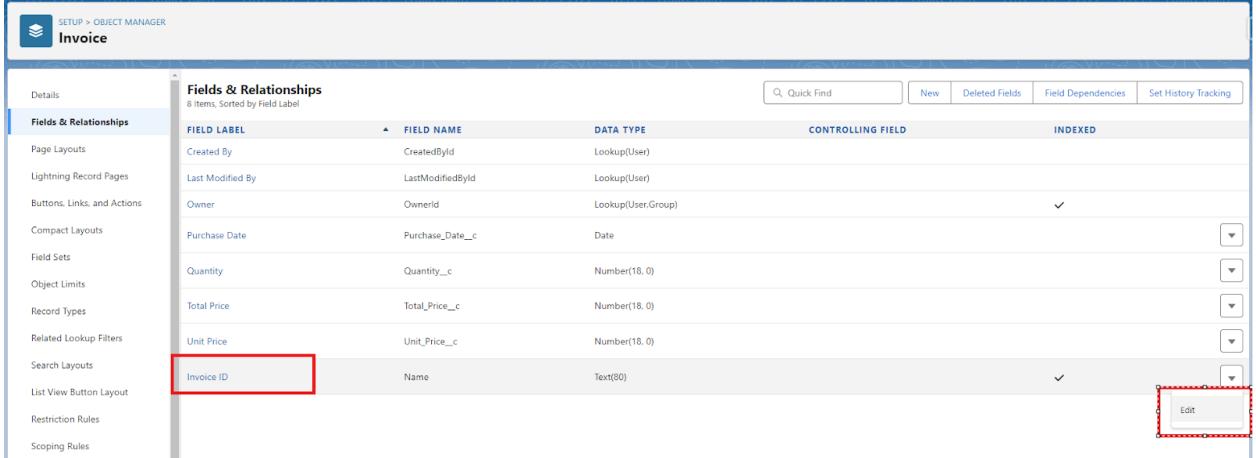
5. click “Check Syntax” and Next >> Next >> Save.

## Step :13

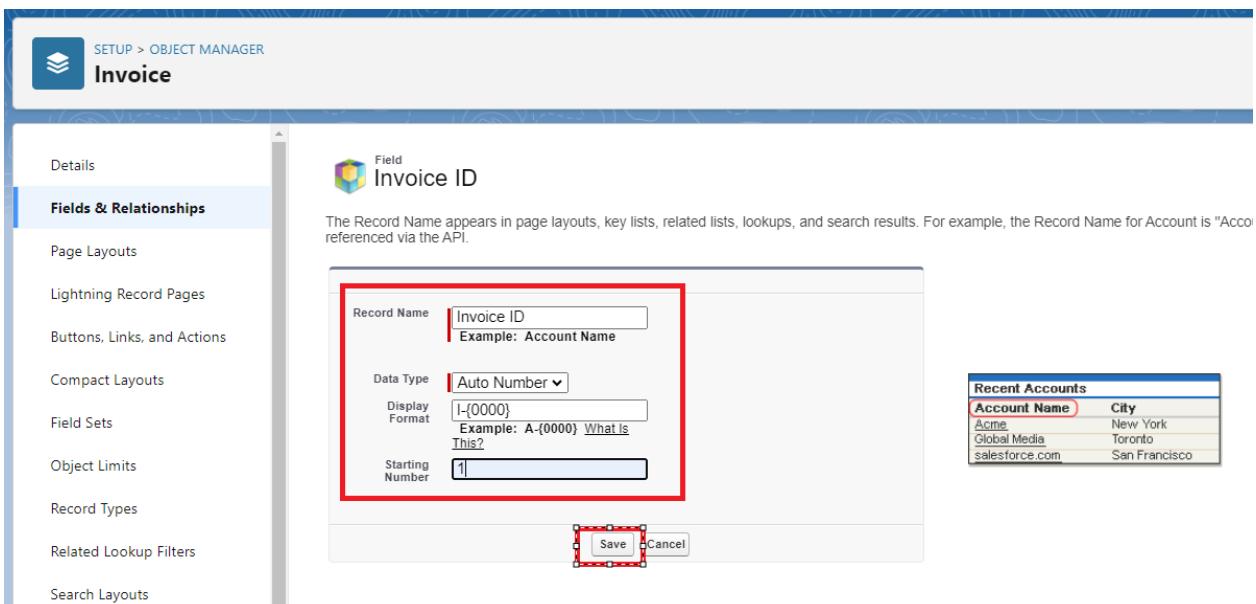
### Updating field in Invoice Object

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.



The screenshot shows the Salesforce Setup interface for the Invoice object. The 'Fields & Relationships' tab is active. A specific row for the 'Invoice ID' field is highlighted with a red box. In the bottom right corner of the list area, there is an 'Edit' button, also highlighted with a red box.



The screenshot shows the 'Edit Field' dialog for the 'Invoice ID' field. The 'Record Name' is set to 'Invoice ID'. The 'Data Type' is set to 'Auto Number' with a display format of 'I-{0000}' and a starting number of '1'. The 'Save' button is highlighted with a red box. On the right side, there is a sidebar titled 'Recent Accounts' showing a list of 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:-

Click Save.

## Step :14

### Creating Remaining Fields in Objects

Now create the remaining fields using the data types mentioned.

S.no	Object name	Fields				
1	Invoice	<table border="1"> <tr> <td><b>Field Name</b></td><td>Opportunity</td></tr> <tr> <td><b>Data type</b></td><td>Master Detail relationship Object : Opportunity</td></tr> </table>	<b>Field Name</b>	Opportunity	<b>Data type</b>	Master Detail relationship Object : Opportunity
<b>Field Name</b>	Opportunity					
<b>Data type</b>	Master Detail relationship Object : Opportunity					

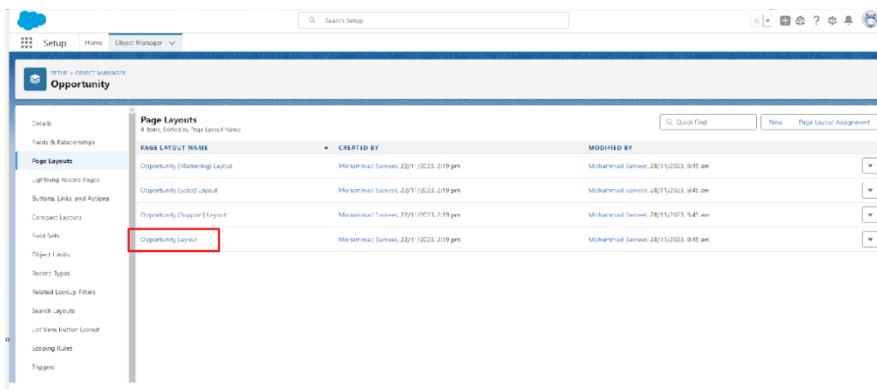
## Step :15

### 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

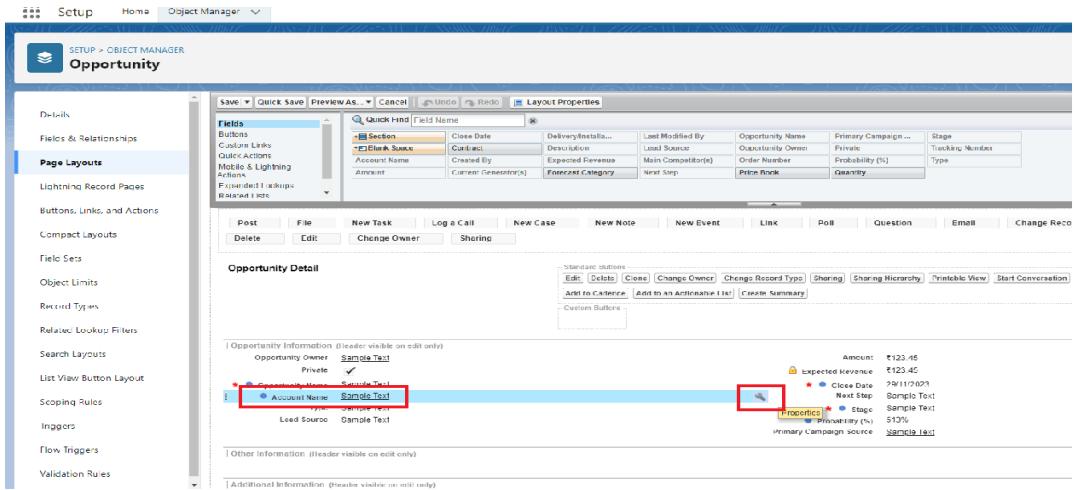
Step 2: Click on Page Layouts, Click on ‘Opportunity Layouts’.



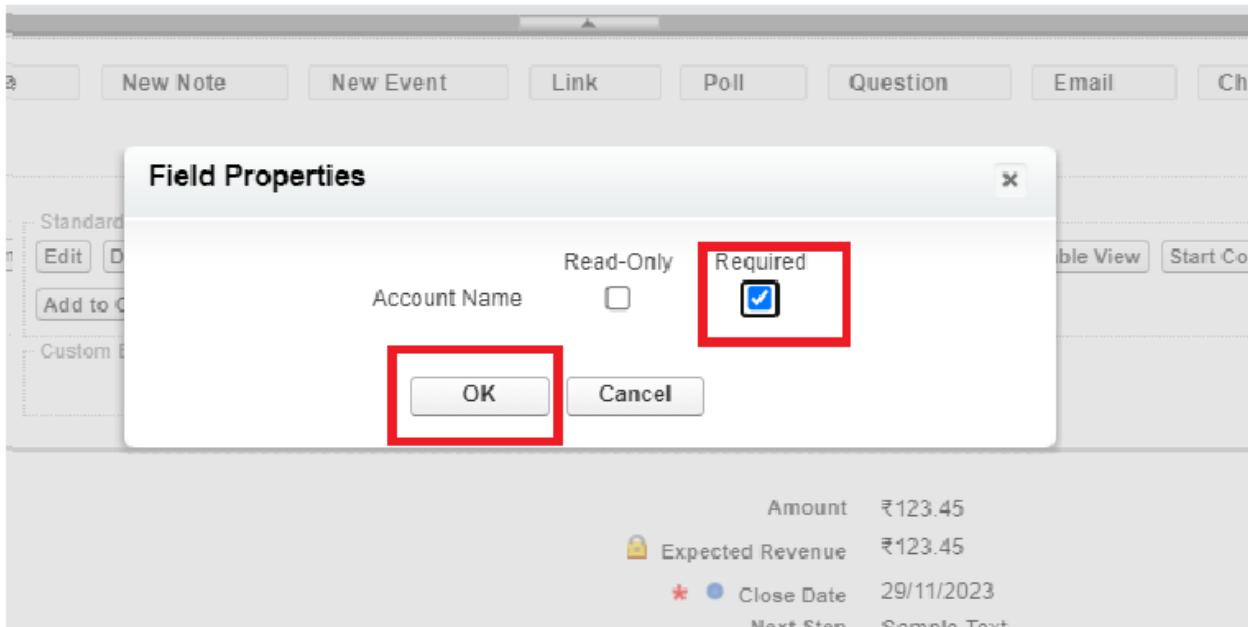
The screenshot shows the Salesforce Object Manager interface for the Opportunity object. The left sidebar lists various configuration tabs like Details, Fields & Relationships, Page Layouts, Buttons, Links, and Actions, Concept Layouts, Field Sets, Object Tools, Record Types, Related Lookup Filters, Search Layouts, List View Behavior Layout, Logging Rules, and Triggers. The 'Page Layouts' tab is selected. The main content area displays a table titled 'Page Layouts' with three rows. The first row is 'Opportunity (Marketing) Layout', the second is 'Opportunity (Sales) Layout', and the third is 'Opportunity (Standard) Layout'. The 'Opportunity Layout' is highlighted with a red box. The table includes columns for 'Page Layout Name', 'Created By', and 'Modified By'.

Page Layout Name	Created By	Modified By
Opportunity (Marketing) Layout	Mohammed Samiex, 23/11/2023, 9:15 pm	Mohammed Samiex, 28/11/2023, 9:15 am
Opportunity (Sales) Layout	Mohammed Samiex, 23/11/2023, 9:19 pm	Mohammed Samiex, 28/11/2023, 9:15 am
Opportunity (Standard) Layout	Mohammed Samiex, 23/11/2023, 9:19 pm	Mohammed Samiex, 28/11/2023, 9:15 am
Opportunity Layout	Mohammed Samiex, 23/11/2023, 9:19 pm	Mohammed Samiex, 28/11/2023, 9:15 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 Salesforce Object Manager interface for the 'Opportunity' object. On the left sidebar, under 'Page Layouts', the 'Opportunity' layout is selected. The main area displays the 'Opportunity Detail' section with various fields like 'Account Name', 'Amount', and 'Close Date'. A red box highlights the 'Account Name' field on the layout.



The screenshot shows the 'Field Properties' dialog box for the 'Account Name' field. The 'Required' checkbox is checked and highlighted with a red box. The 'OK' button is also highlighted with a red box. In the background, the Opportunity detail page is visible with the 'Account Name' field highlighted.

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

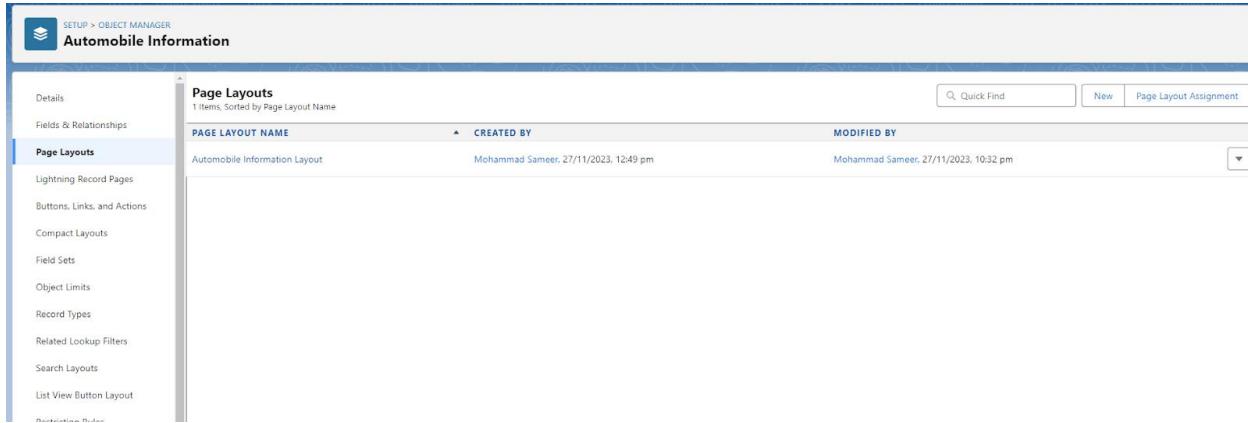
Step 5: Click on Save.

## Step :16

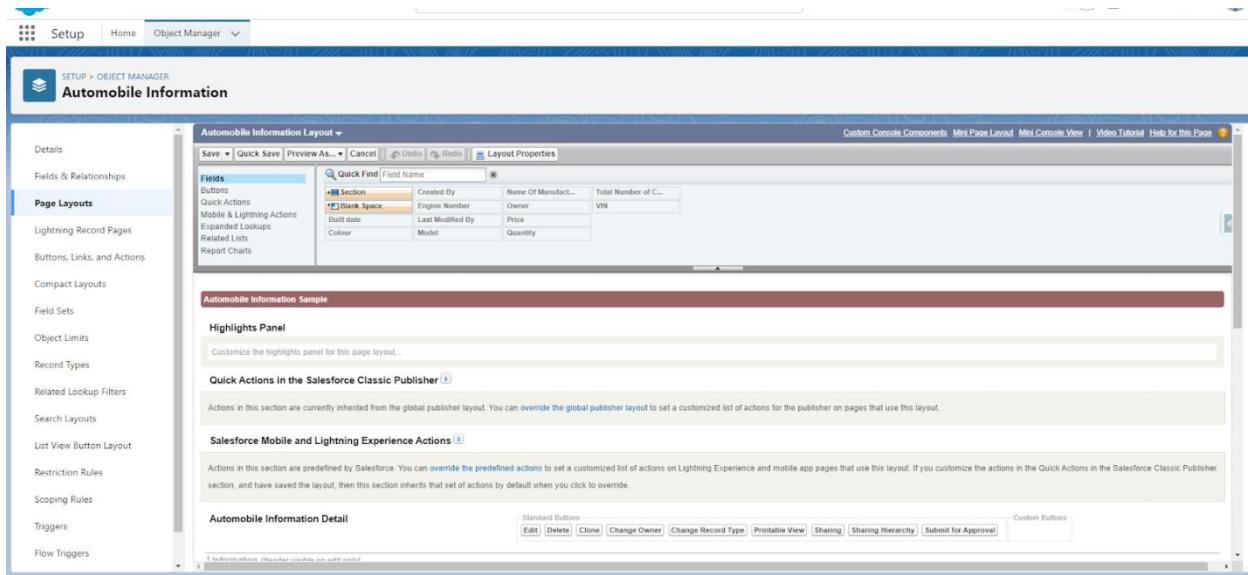
### Edit the Page layout for Automobiles Information

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’.

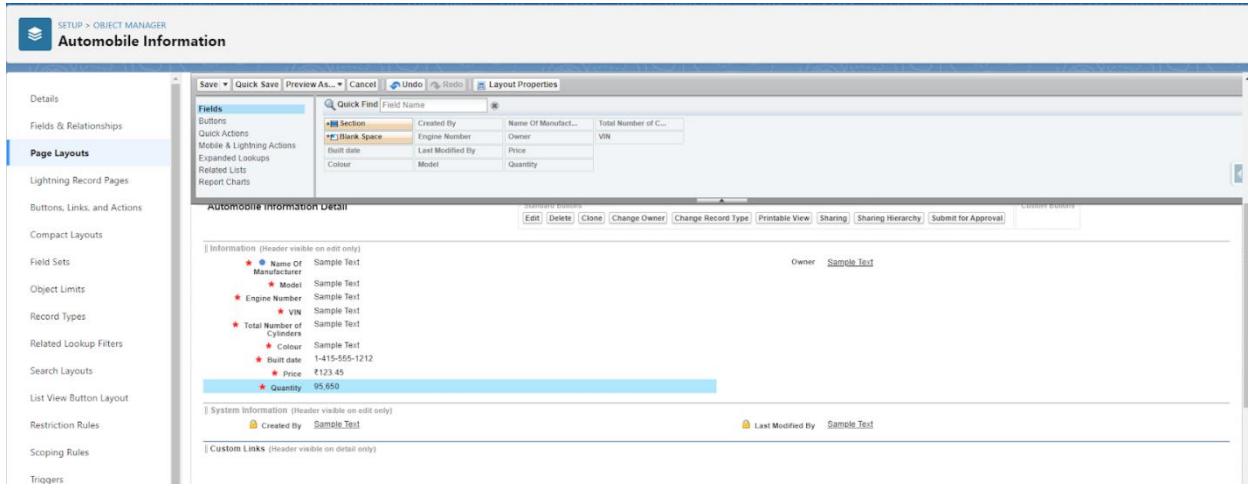


The screenshot shows the Salesforce Setup interface under the Object Manager for the 'Automobile Information' object. The left sidebar is collapsed, and the main area displays the 'Page Layouts' section. A single item is listed: 'Automobile Information Layout'. The details show it was created by Mohammad Sameer on 27/11/2023 at 12:49 pm and modified by Mohammad Sameer on 27/11/2023 at 10:32 pm. The page layout itself is not visible in this view.



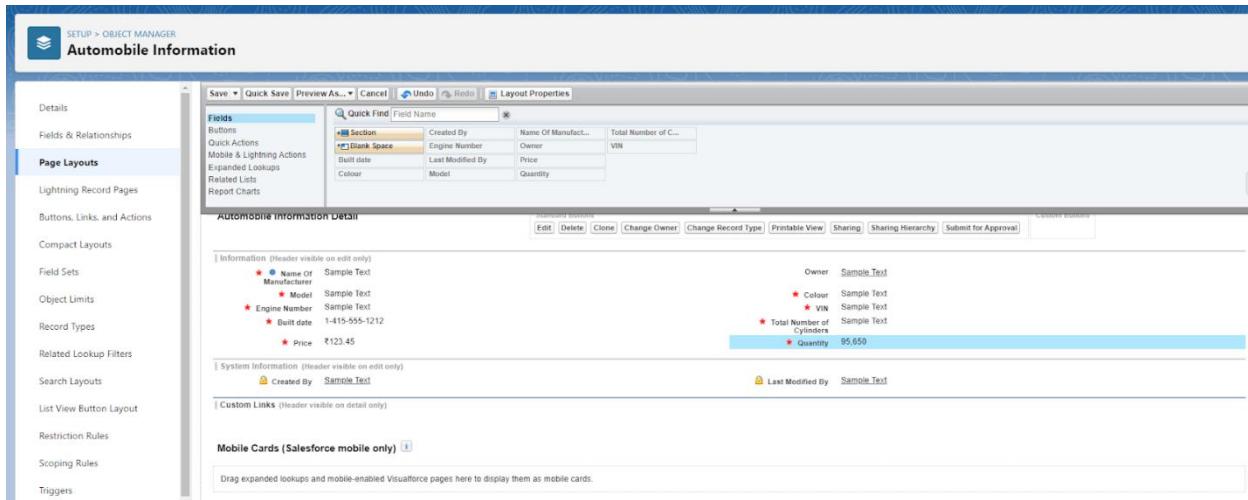
This screenshot shows the detailed view of the 'Automobile Information Layout' within the Setup interface. The left sidebar is expanded, showing various layout configuration options like Details, Fields & Relationships, Page Layouts, and Buttons, Links, and Actions. The 'Fields' section is selected, displaying a list of fields: Buttons, Quick Actions, Model, Lightning Actions, Expanded Lookups, Related Lists, and Report Charts. The main pane shows the layout's structure with sections for 'Fields' (containing a 'Blank Space' component), 'Automobile Information Sample' (with 'Highlights Panel' and 'Quick Actions in the Salesforce Classic Publisher'), and 'Automobile Information Detail' (with standard buttons like Edit, Delete, Clone, etc.).

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 'Page Layouts' section of the Salesforce setup for the 'Automobile Information' object. The layout consists of several sections: 'Fields' (Buttons, Quick Actions, Mobile & Lightning Actions, Expanded Lookups, Related Lists, Report Charts), 'Information' (Header visible on edit only), 'System Information' (Header visible on edit only), and 'Custom Links' (Header visible on detail only). The 'Information' section contains fields for Name Of Manufacturer, Model, Engine Number, VIN, Total Number of Cylinders, Colour, Built date, Price, and Quantity. The 'System Information' section shows Created By and Last Modified By. The 'Custom Links' section is currently empty.

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



The screenshot shows the 'Page Layouts' section of the Salesforce setup for the 'Automobile Information' object. The layout includes sections for Fields, Buttons, Quick Actions, Mobile & Lightning Actions, Expanded Lookups, Related Lists, and Report Charts. The main detail page shows fields for Name Of Manufacturer, Model, Engine Number, VIN, Total Number of Cylinders, Colour, Built date, Price, and Quantity. A new section titled 'Mobile Cards (Salesforce mobile only)' is added at the bottom, which contains a note: 'Drag expanded lookups and mobile-enabled Visualforce pages here to display them as mobile cards.'

Step 5 : Click on Save.

## Step :17

### Opportunity Automobile quantity

**Use Case : Whenever Opportunity Closed won Then 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 “Opportunity Handler Class ”.

```

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 Opportunity Handler Class {

    public static void Opportunity Automobile Quantity(List<Opportunity> Lst Opportunity, Map<Id, Opportunity>
Old Map Opportunity){

        set<Id> opportunity Ids = new set<Id>();
        for(Opportunity Opp : Lst Opportunity){
            if(Opp. Stage Name =='Closed Won'){
                opportunity Ids.add(Opp.Id);
            }
        }
    }
}

```

```

}

Map< Id, Opportunity _ Automobile __c> lst Opportunity Automobile =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: opportunity Ids]);

set<Id> Auto Information Ids = new set<Id>();
for(Opportunity _Automobile__c Opp Auto: lst Opportunity Automobile.values()){
    if(Opp Auto .Automobile __c != null){
        Auto Information Ids.add(OppAuto.Automobile__c);
    }
}
List<Automobile _Information __c> lst Automobile Infomation = new List<Automobile_Information__c>();
Map<Id,Automobile Information __c> Map Automobile Information = New Map<Id,Automobile_Information
__c>([SELECT Quantity __c, Price __c, Name, Id FROM Automobile _Information __c WHERE Id IN: Auto
InformationIds]);
For(Opportunity_Automobile__c AutoOpp : lst Opportunity Automobile.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;
}
}

}
}

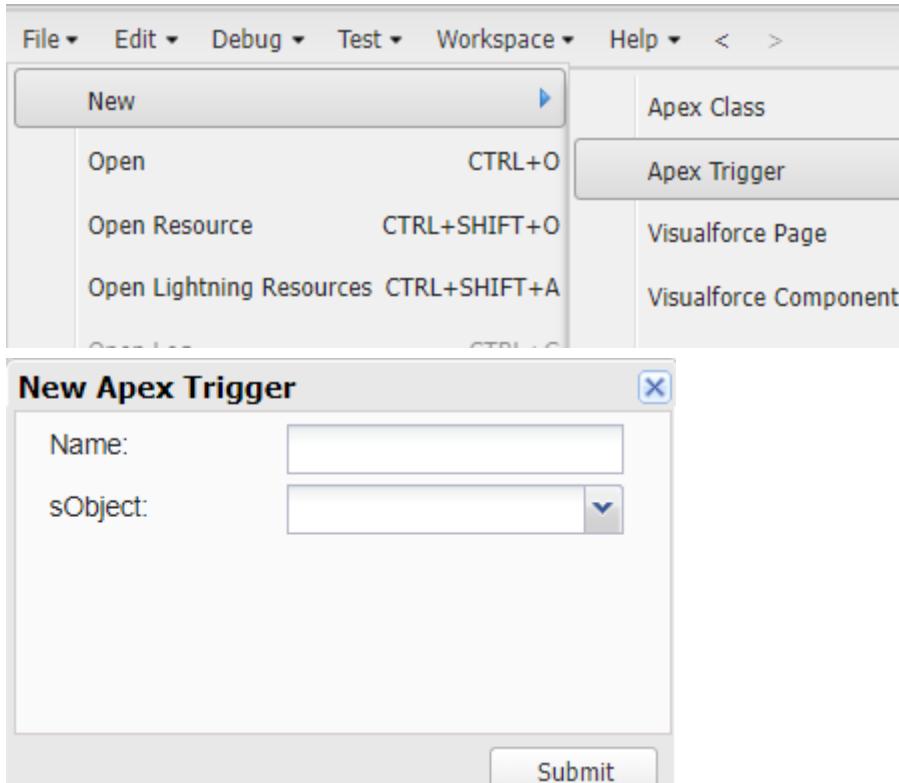
```

### 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 :

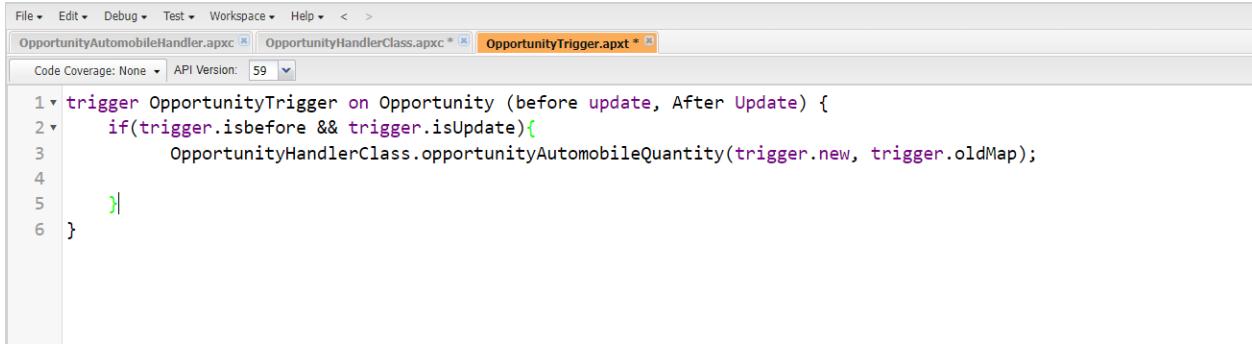
```
Trigger [trigger name] on [object name]( Before/After event){
```

```
//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.



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

**Code:**

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

**Step :18****Opportunity-Automobile Error**

**Use Case : 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 “Opportunity Automobile Handler ”.

```

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                  OppAutomobileaddError('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(OppAutomobile.Automobile__c).Quantity__c );
            }
        }
    }
}

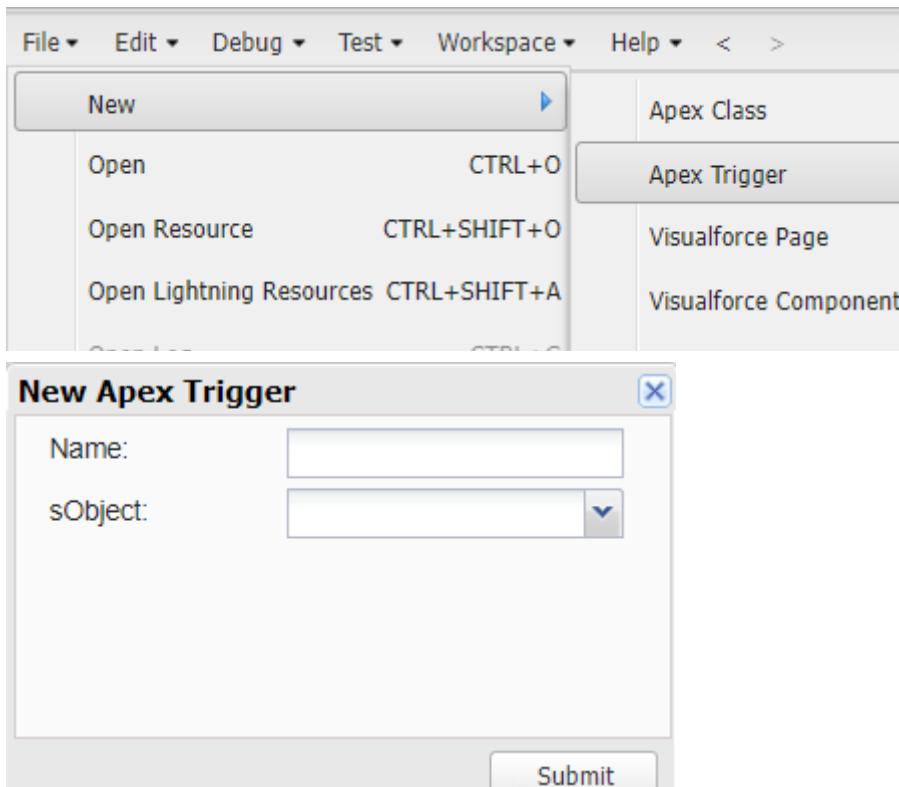
```

### Trigger Handler :

How to create a new trigger :

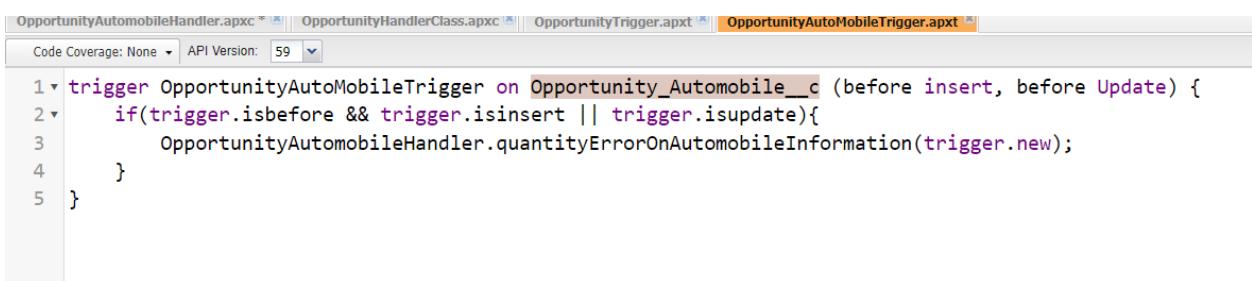
1. While still in the trailhead account, navigate to the gear icon in the top right corner.
2. Click on developer console and you will be navigated to a new console window.
3. Click on the File menu in the toolbar, and click on new? Trigger.

4. Enter the trigger name and the object to be triggered.
5. Name : Opportunity Auto Mobile Trigger
6. Subject : 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.new);
```

```

    }
}
```

## Step :19

### Invoice Creation Trigger

**Use Case : 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 Opportunity Trigger.

Open

Entity Type	Entities	Related			
	Name	Namespace	Name	Extent	Direction
Entity Type					
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     IF(trigger.isafter && trigger.isupdate){
6         InvoiceCreation.OpportunityClosedwonInvoiceGeneration(trigger.new, trigger.oldMap);
7     }
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);
    }
}
```

## Step :20

### Check contact role

**Use Case : 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.');
                }
            }
        }
    }
}

```

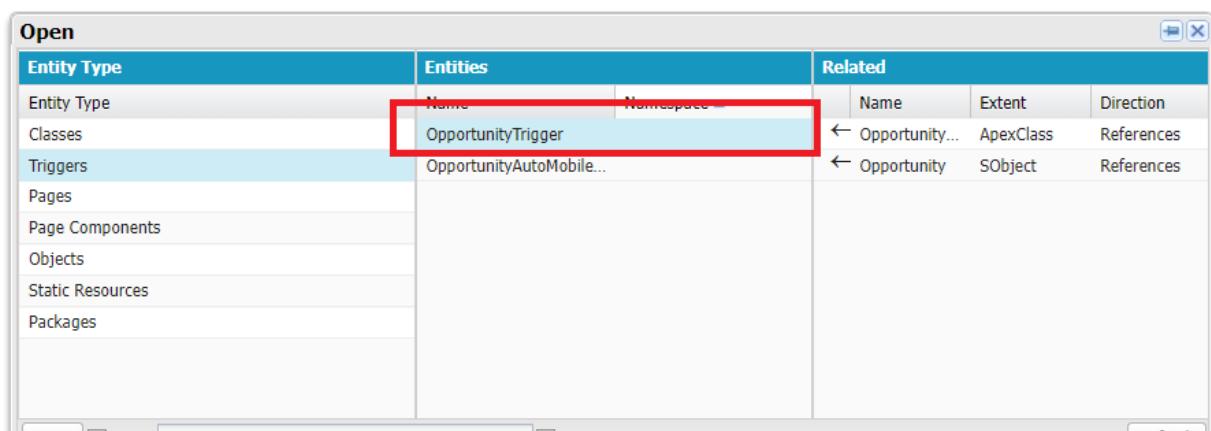
```

        }
    }
}
}
}
}
```

### **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 Opportunity Trigger.



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);  
    }  
}
```

## Step :21

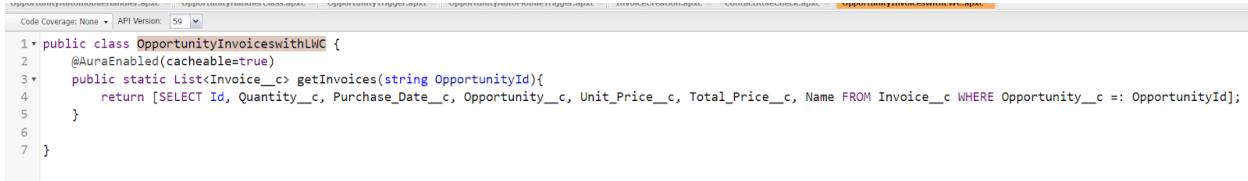
### Create Apex Class to Get Invoices

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.

**In the toolbar, you can see FILE. Install Salesforce CLI**

- 4.
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
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];
    }
}
```

## Step :22

### 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    
```

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

## Step :23

### 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.

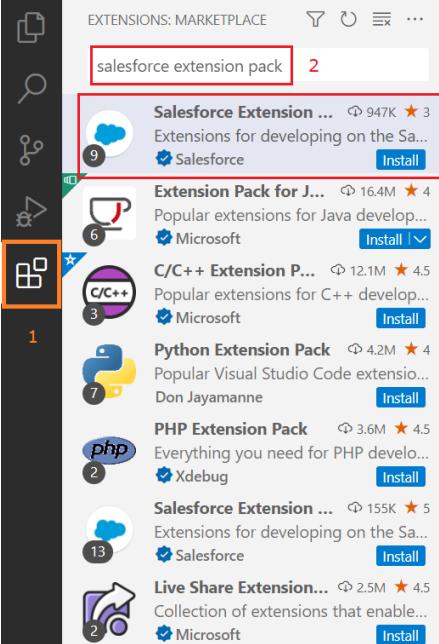
The following instructions are for Windows OS. Other operating systems may have slightly different steps.

## Step :24

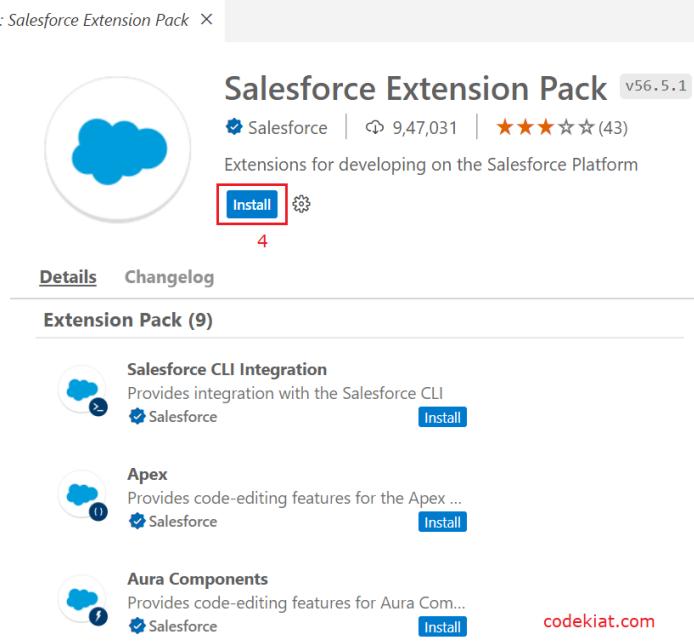
### Install the Salesforce Extension Pack

In the VS Code,

1. go to extensins (1) as shown in the image below.
2. Search with the Salesforce extension pack (2) as shown in the image below.
3. select Salesforce Extension Pack from the list (3) as shown in the image below.
4. Click the Install button (4) as shown in the image below.

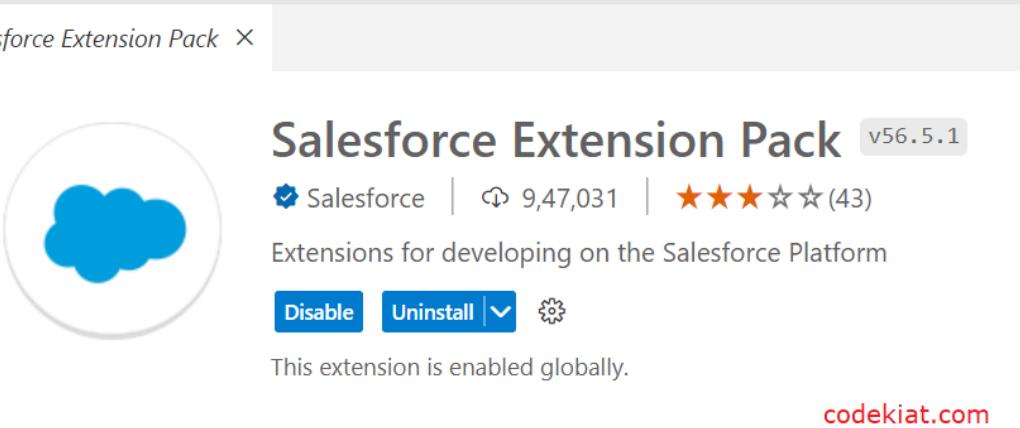


The screenshot shows the VS Code Marketplace interface. A sidebar on the left has a 'Extensions' icon highlighted with a red box (step 1). In the search bar at the top, 'salesforce extension pack' is typed, and the count '2' is shown (step 2). Below the search bar, the 'Salesforce Extension Pack' by Salesforce is listed with a blue 'Install' button (step 3). Other extensions like 'Extension Pack for Java' and 'C/C++ Extension Pack' are also visible.



The right side shows the detailed view of the 'Salesforce Extension Pack' version v56.5.1. It includes a large cloud icon, developer information (Salesforce, 9,47,031 installs, 4.5 stars), and a prominent blue 'Install' button. Below the main title, there's a section titled 'Extension Pack (9)' which lists three sub-components: 'Salesforce CLI Integration', 'Apex', and 'Aura Components', each with its own 'Install' button. The URL 'codekiat.com' is visible at the bottom right.

The extension pack is installed successfully



This screenshot shows the same extension details page as above, but now the 'Uninstall' button is highlighted with a red box (step 4). The page title is 'Extension: Salesforce Extension Pack'. It displays the extension's name, developer (Salesforce), popularity (9,47,031 installs, 4.5 stars), and a brief description: 'Extensions for developing on the Salesforce Platform'. Below the main title, there are buttons for 'Disable', 'Uninstall' (highlighted with a red box), and a gear icon. A message at the bottom states 'This extension is enabled globally.' The URL 'codekiat.com' is visible at the bottom right.

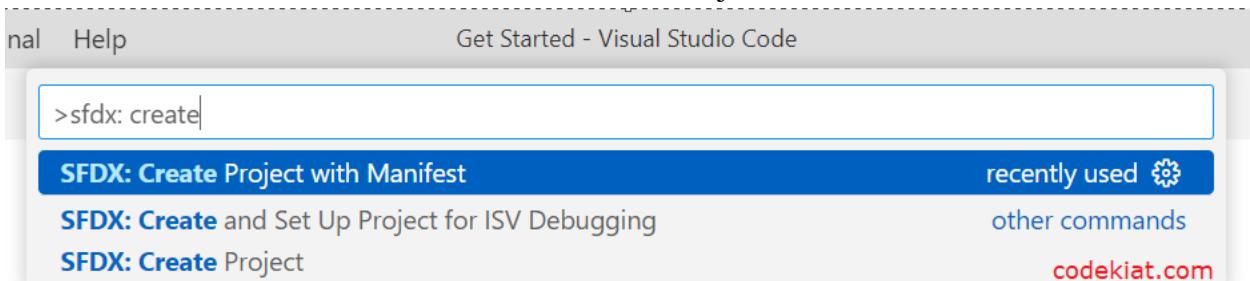
Install the Salesforce Extension Pack

## Step :24

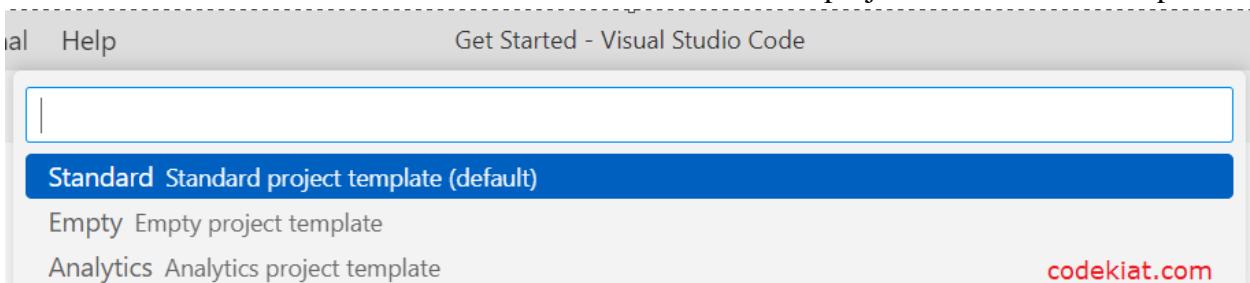
### Create a project in VS Code

1. Press CTRL + SHIFT + P, type sfdx: create

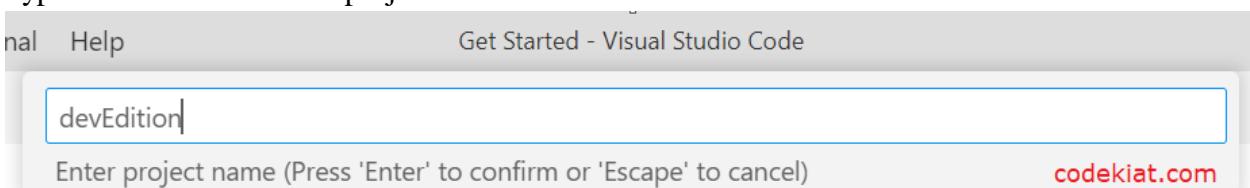
2. select SFDX: Create Project with Manifest



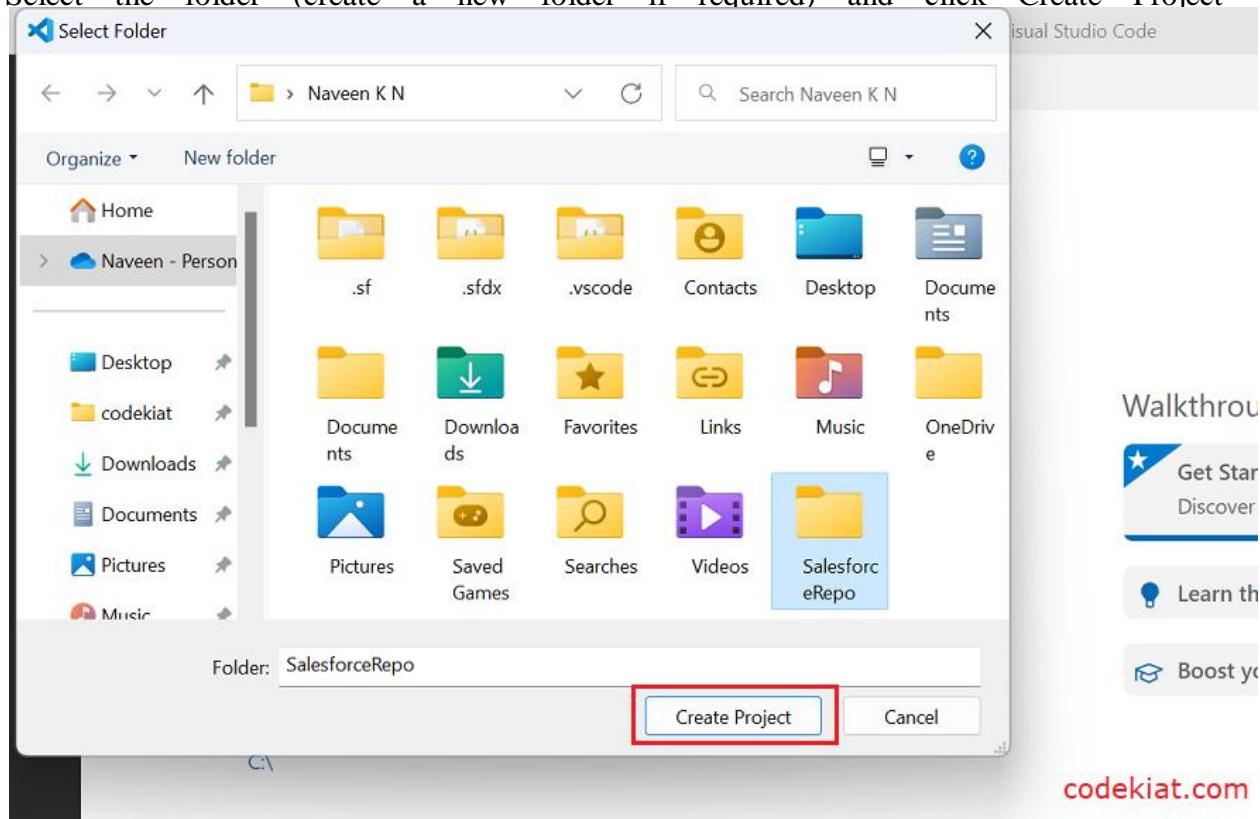
3. Select the Standard project template



4. Type a project name and Click Enter.

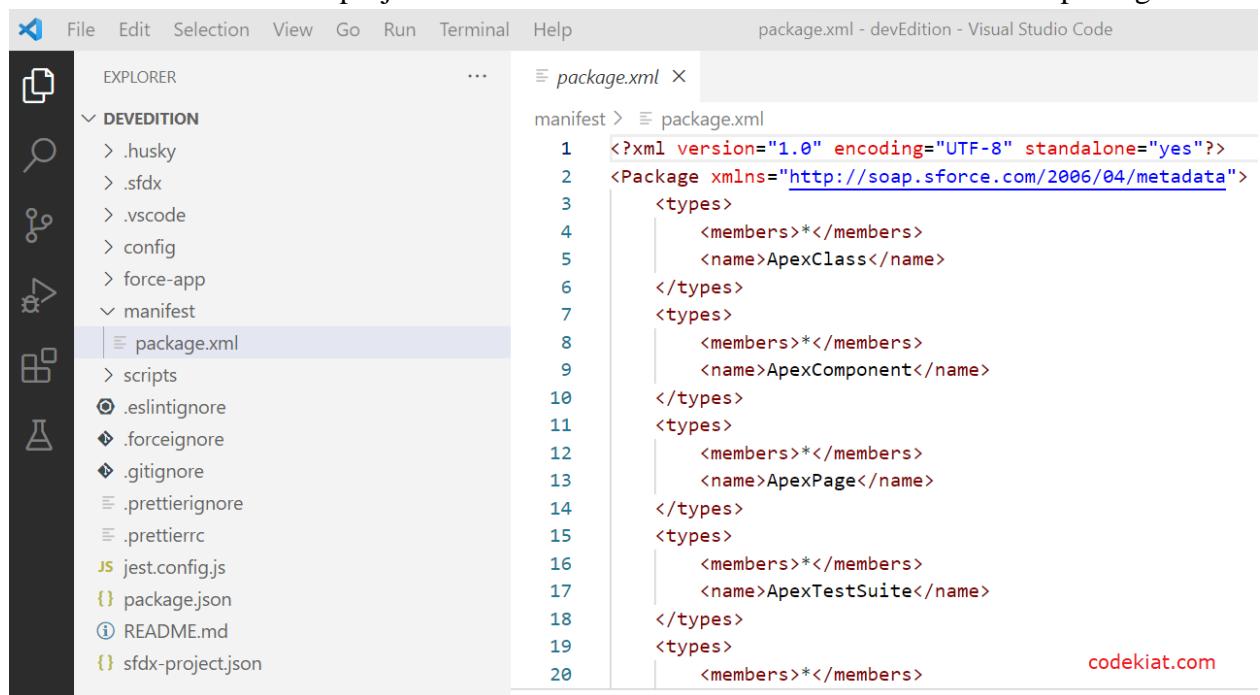


5. Select the folder (create a new folder if required) and click Create Project



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

6. The new project is created with package.xml



Default Package.xml contains various metadata types. I have updated Package.xml as shown below as we deal only with LWC in this article.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members>*

```

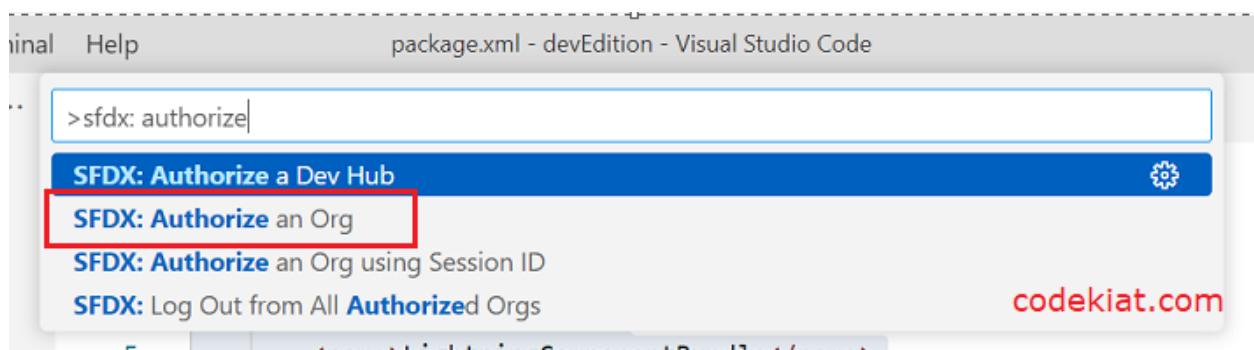
## Step :25

### Authorize an org

**Establish a connection between the local project and the Salesforce instance to retrieve and deploy the components.**

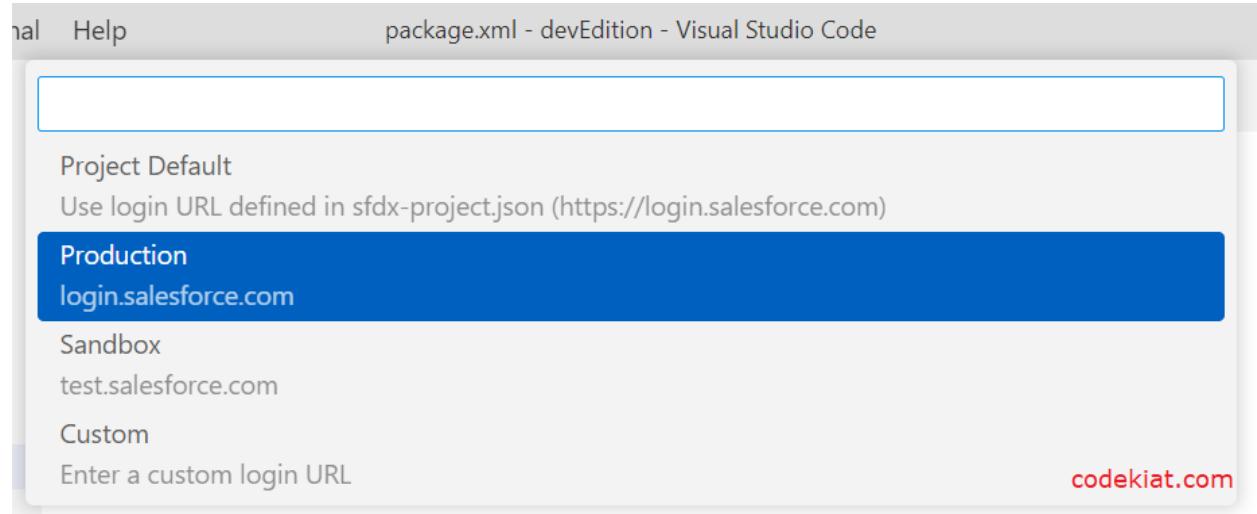
1. Press CTRL + SHIFT + P, type sfdx: authorize.

2. select SFDX: Authorize an Org from the list

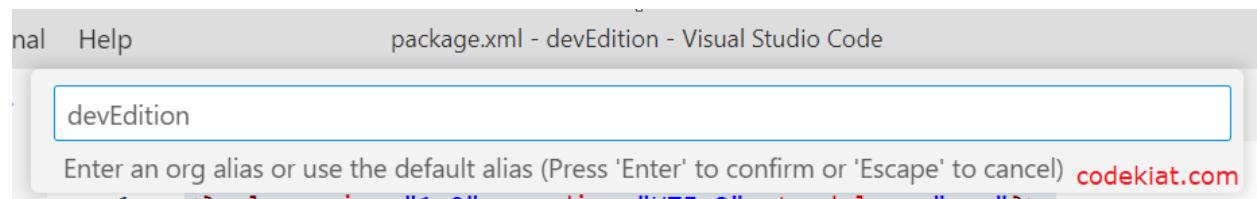


3. Choose your Salesforce instance.  
For developer edition and production instances select

Production.



4. For this demonstration, I used the developer edition, hence it is Production.
5. Give a project name and press Enter



6. The Salesforce login page opens in the browser.

7. Enter the credentials and click Log In

 [login.salesforce.com/?startURL=%2Fsetup%2Fsecur%2FRemoteAccess](https://login.salesforce.com/?startURL=%2Fsetup%2Fsecur%2FRemoteAccess)

codekiat.com



The screenshot shows the Salesforce login interface. At the top, there's a large blue cloud-shaped logo with the word "salesforce" in white. Below it is a form with fields for "Username" and "Password". To the right of the "Username" field, it says "1 Saved Username". A large blue "Log In" button is centered below the password field. Below the form, there's a "Remember me" checkbox followed by a link "Forgot Your Password?" and another link "Use Custom Domain".

Username [1 Saved Username](#)

Password

Remember me

[Forgot Your Password?](#) [Use Custom Domain](#)

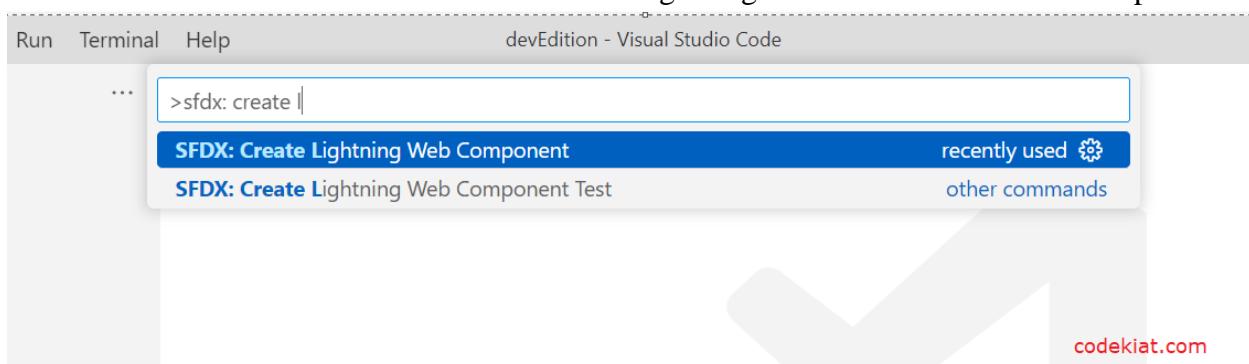
8. It will be successfully authorized.

## Step :26

### Create Lightning Web Component

**XML File :**

- In the VS Code, press CTRL + SHIFT + P, type sfdx: create lightning in the search bar, and select SFDX: Create Lightning Web Component



- Give the name "InvoiceOpportunity" and press Enter.



```


<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordAction</target>
        <target>lightning__RecordPage</target>
    </targets>
</LightningComponentBundle>


```

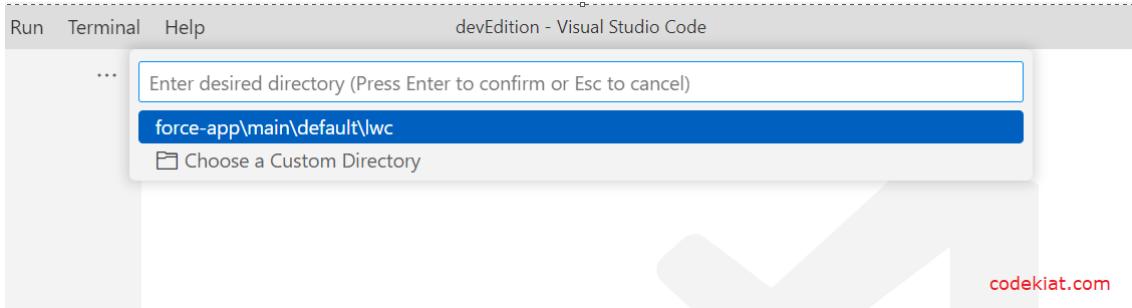
The screenshot shows the Visual Studio Code editor with a dark theme. The top navigation bar includes 'File', 'Edit', 'View', 'Go', 'Run', and a '...' button. The main area shows a file named 'InvoiceOpportunity.js-meta.xml'. The code in the editor is:

```

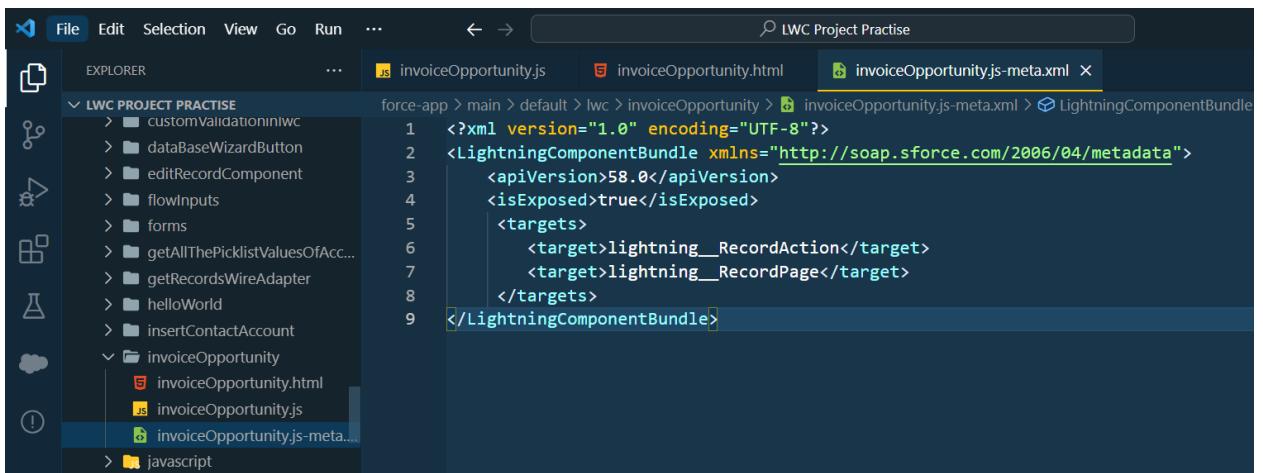
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordAction</target>
        <target>lightning__RecordPage</target>
    </targets>
</LightningComponentBundle>

```

3. Choose the directory.



4. LWC is created successfully.



5. Copy and paste the below-mentioned code in the InvoiceOpportunity.js-meta.xml and update the apiVersion tag with the latest API version.

#### XML File Code:

```

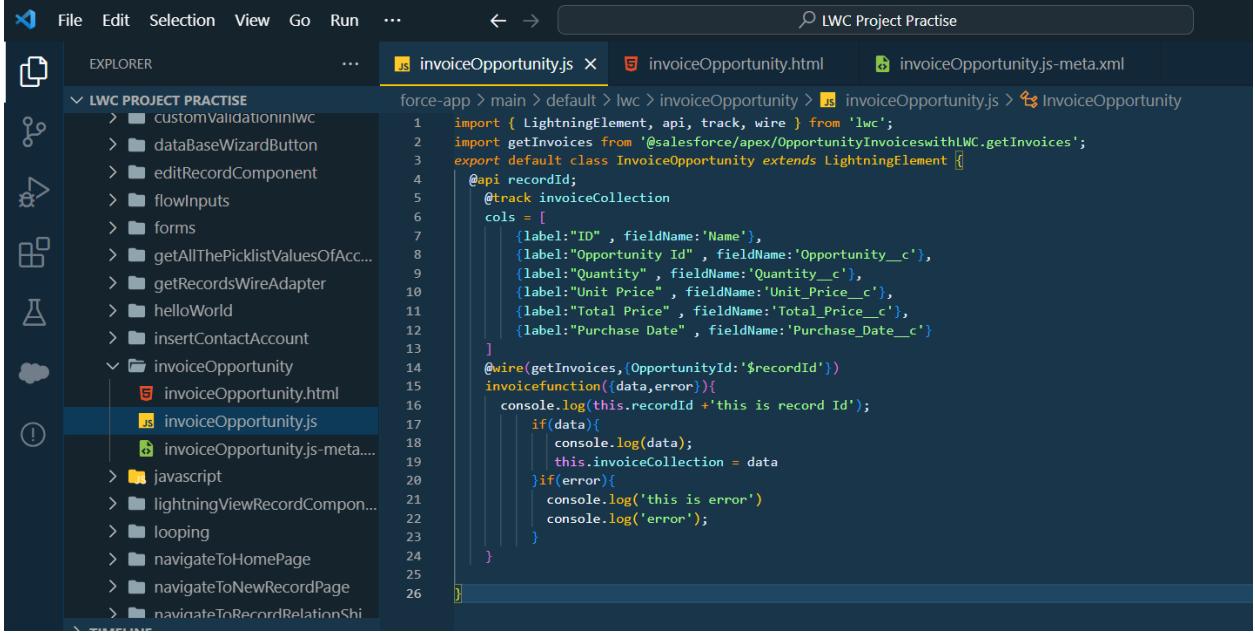
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordAction</target>
        <target>lightning__RecordPage</target>
    </targets>

```

</LightningComponentBundle>

### JS File :

1. Copy and paste the below-mentioned code in the InvoiceOpportunity.js and update the apiVersion tag with the latest API version.



The screenshot shows the LWC Project Practise interface. The left sidebar displays a file tree under 'LWC PROJECT PRACTISE'. The 'invoiceOpportunity' folder is expanded, showing 'invoiceOpportunity.html' and 'invoiceOpportunity.js'. The 'invoiceOpportunity.js' file is selected and open in the main editor area. The code in the editor is:

```

import { LightningElement, api, track, wire } from 'lwc';
import getInvoices from '@salesforce/apex/OpportunityInvoiceswithLWC.getInvoices';
export default class InvoiceOpportunity extends LightningElement {
    @api recordId;
    @track invoiceCollection
    cols = [
        {label:"ID" , fieldName:'Name'},
        {label:"Opportunity Id" , fieldName:'Opportunity__c'},
        {label:"Quantity" , fieldName:'Quantity__c'},
        {label:"Unit Price" , fieldName:'Unit_Price__c'},
        {label:"Total Price" , fieldName:'Total_Price__c'}
    ]
    @wire(getInvoices,{OpportunityId:$recordId})
    invoicefunction({data,error}){
        console.log(this.recordId +'this is record Id');
        if(data){
            console.log(data);
            this.invoiceCollection = data
        }if(error){
            console.log('this is error')
            console.log(error);
        }
    }
}

```

### JS File Code :

```

import { LightningElement, api, track, wire } from 'lwc';
import getInvoices from '@salesforce/apex/OpportunityInvoiceswithLWC.getInvoices';
export default class InvoiceOpportunity extends LightningElement {

    @api recordId;

    @track invoiceCollection

    cols = [

        {label:"ID" , fieldName:'Name'},

        {label:"Opportunity Id" , fieldName:'Opportunity__c'},

        {label:"Quantity" , fieldName:'Quantity__c'},

        {label:"Unit Price" , fieldName:'Unit_Price__c'},

        {label:"Total Price" , fieldName:'Total_Price__c'},
    ]
}

```

```

        {label:"Purchase Date" , fieldName:'Purchase_Date__c'}
    ]
    @wire(getInvoices,{OpportunityId:$recordId})
    invoicefunction({data,error}){  

        console.log(this.recordId +'this is record Id');  

        if(data){  

            console.log(data);  

            this.invoiceCollection = data  

        }if(error){  

            console.log('this is error')  

            console.log(error);  

        }  

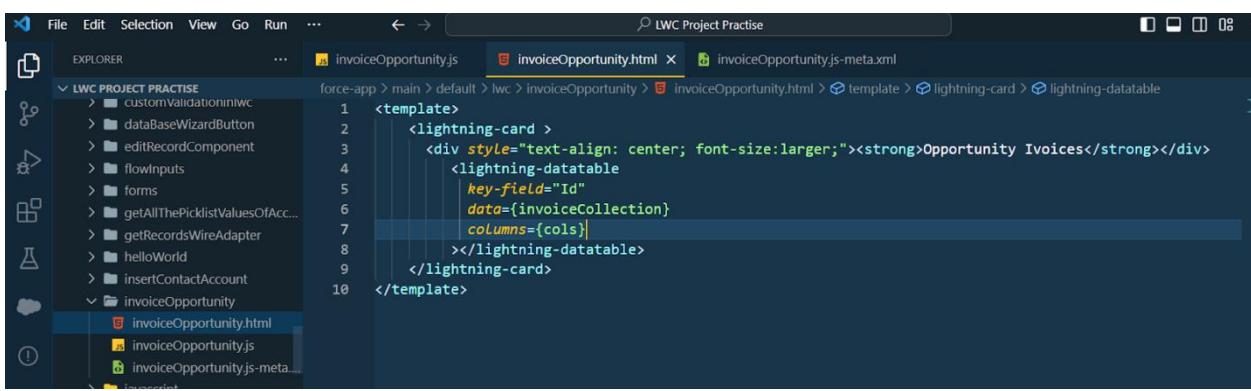
    }  

}

```

### HTML File :

1. Copy and paste the below-mentioned code in the InvoiceOpportunity.html and update the apiVersion tag with the latest API version.



```

<template>
    <lightning-card>
        <div style="text-align: center; font-size:larger;"><strong>Opportunity Ivoices</strong></div>
        <lightning-datatable
            key-field="Id"
            data={invoiceCollection}
            columns={cols}>
        </lightning-datatable>
    </lightning-card>
</template>

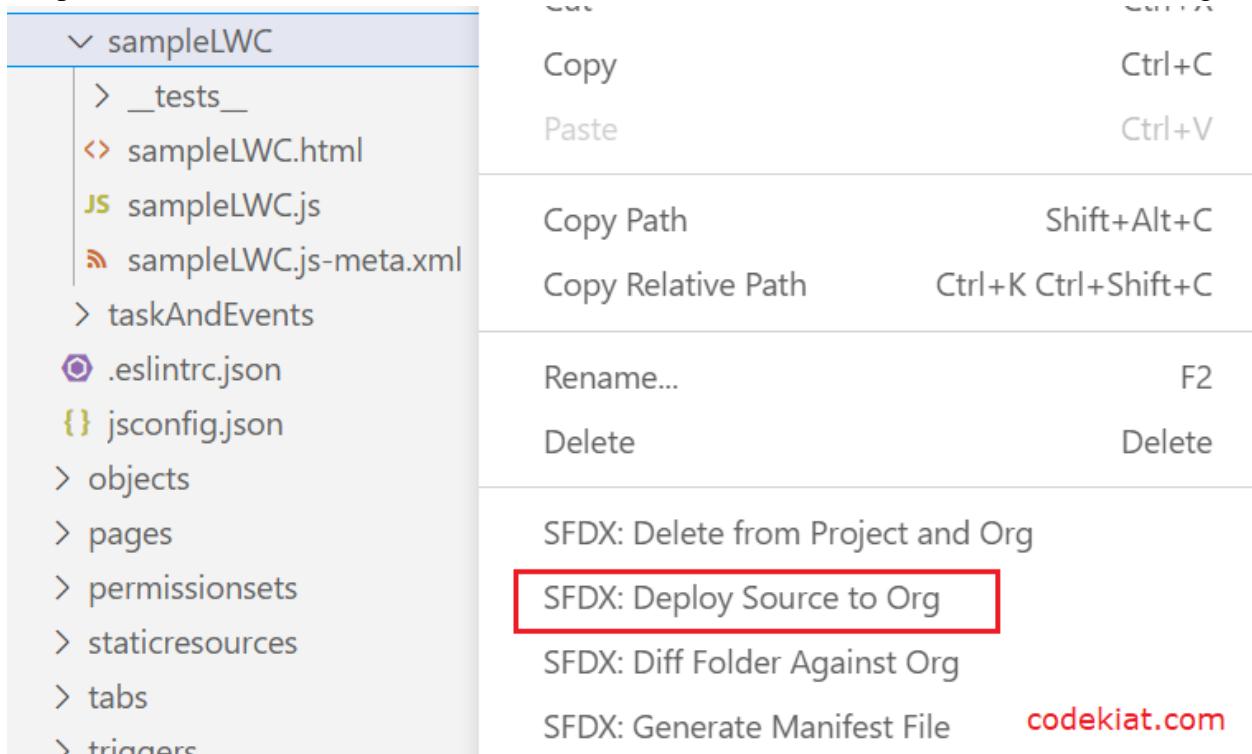
```

### HTML File Code:

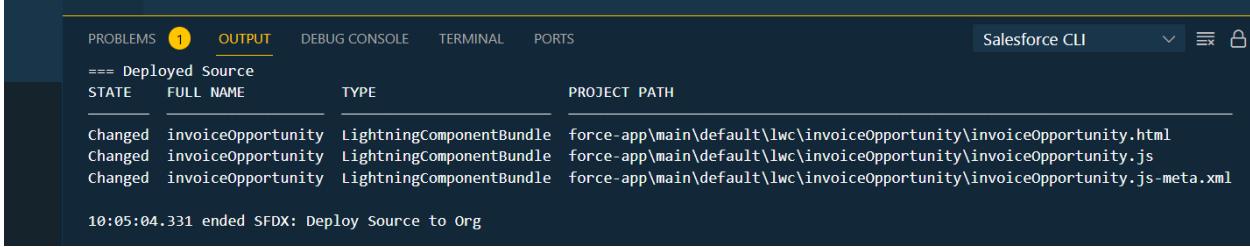
```
<template>
  <lightning-card >
    <div style="text-align: center; font-size:larger;"><strong>Opportunity Ivoices</strong></div>
    <lightning-datatable
      key-field="Id"
      data={invoiceCollection}
      columns={cols}
    ></lightning-datatable>
  </lightning-card>
</template>
```

### Deploy Component:

1. Right-click on the component folder, and select SFDX: Deploy Source to Org to deploy the component to the org.



- Once the deployment is complete, you will see the below-highlighted message in the output tab



The screenshot shows the Salesforce CLI interface with the 'OUTPUT' tab selected. It displays deployment results for a component named 'invoiceOpportunity'. The table includes columns for STATE, FULL NAME, TYPE, and PROJECT PATH. The deployment was successful, indicated by 'Changed' status and a green checkmark icon. The full name is 'invoiceOpportunity', type is 'LightningComponentBundle', and the project path is 'force-app\main\default\lwc\invoiceOpportunity\invoiceOpportunity'. The deployment ended at 10:05:04.331.

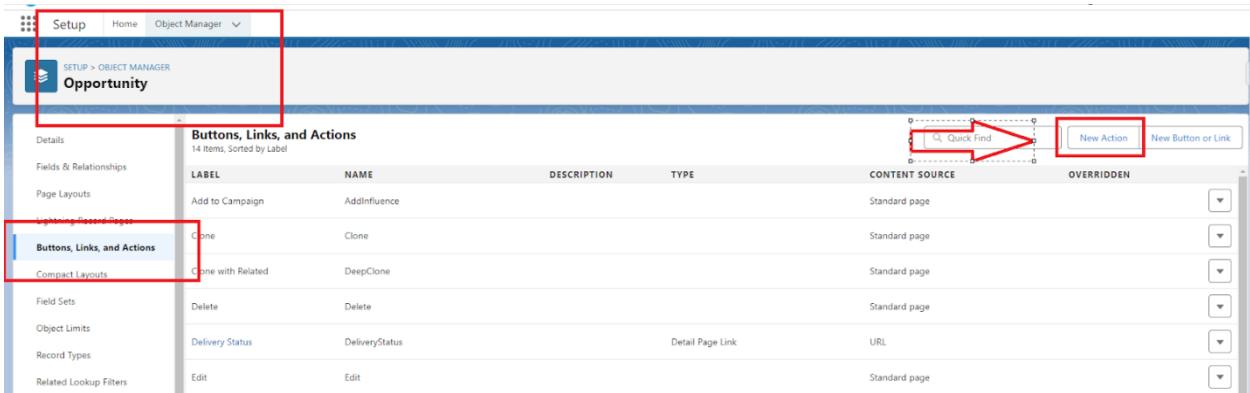
STATE	FULL NAME	TYPE	PROJECT PATH
Changed	invoiceOpportunity	LightningComponentBundle	force-app\main\default\lwc\invoiceOpportunity\invoiceOpportunity.html
Changed	invoiceOpportunity	LightningComponentBundle	force-app\main\default\lwc\invoiceOpportunity\invoiceOpportunity.js
Changed	invoiceOpportunity	LightningComponentBundle	force-app\main\default\lwc\invoiceOpportunity\invoiceOpportunity.js-meta.xml

10:05:04.331 ended SFDX: Deploy Source to org

## Step :27

### Create Button to Add on Opportunity

- To add the newly created component to the view, Go to Salesforce Setup
- Click on Object Manager
- Search Opportunity and Click on it .
- click on Button Links and Action.
- click on the New Action.

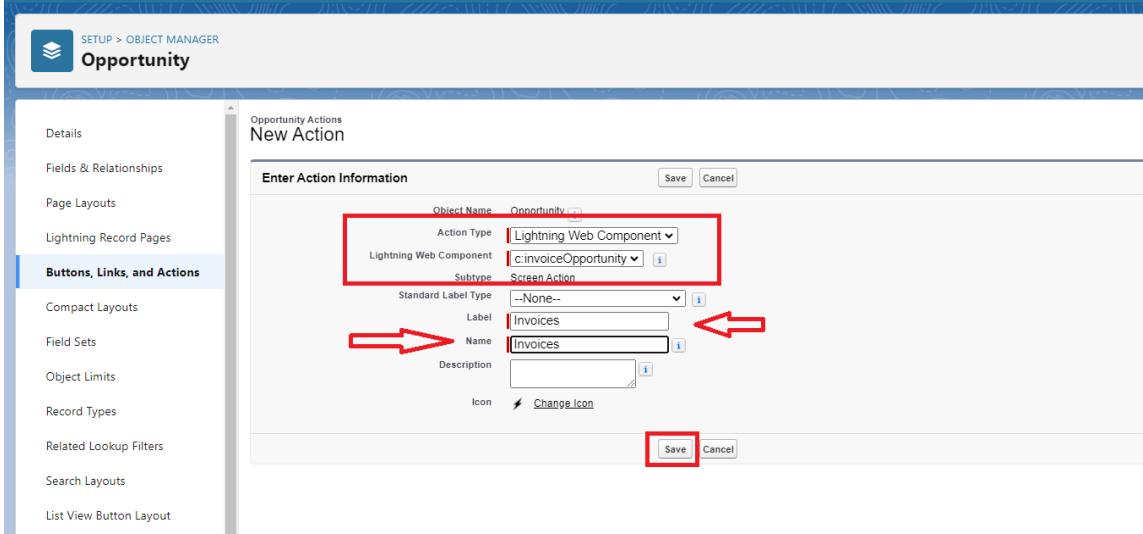


The screenshot shows the Salesforce Setup interface for the 'Opportunity' object. A red box highlights the 'Object Manager' link in the top navigation bar. Another red box highlights the 'Buttons, Links, and Actions' section in the left sidebar. Within this section, a third red box highlights the 'New Action' button in the top right corner of the list table. The table lists various actions such as 'Add to Campaign', 'Clone', 'Delete', etc., with their respective labels, names, descriptions, types, content sources, and overridden status.

LABEL	NAME	DESCRIPTION	TYPE	CONTENT SOURCE	OVERRIDDEN
Add to Campaign	AddInfluence		Standard page		
Clone	Clone		Standard page		
Clone with Related	DeepClone		Standard page		
Delete	Delete		Standard page		
Delivery Status	DeliveryStatus		Detail Page Link	URL	
Edit	Edit		Standard page		

- Select Action type as Lightning Web Component
- Select the InvoiceOpportunity component
  - Label :- Invoices
  - Name :- Invoices

As given on below image



The screenshot shows the 'Opportunity Actions' section of the 'Object Manager'. A new action is being created with the following details:

- Action Name:** Opportunity
- Action Type:** Lightning Web Component
- Lightning Web Component:** c:invoiceOpportunity
- Label:** Invoices
- Name:** Invoices
- Icon:** Change Icon

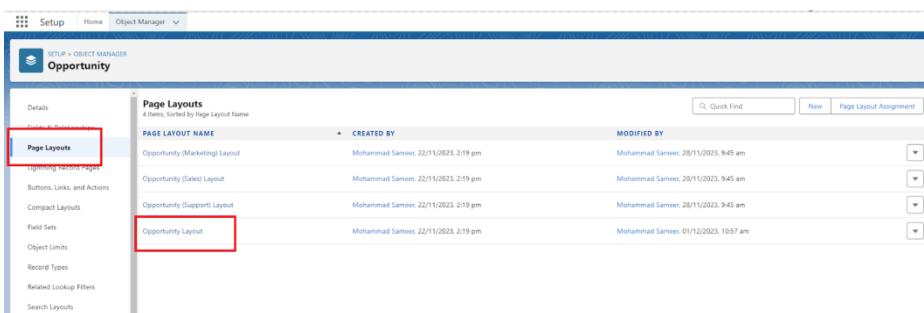
Red boxes highlight the 'Action Type' and 'Lightning Web Component' fields, and red arrows point to the 'Label' and 'Name' fields. The 'Save' button at the bottom right is also highlighted with a red box.

Click on Save and your action Button is Ready.

## Step :28

### Add Invoice Opportunity into Opportunity Record Page

1. On Opportunity Object Manager Click on Page layout.
2. Click on OpportunityLayout.

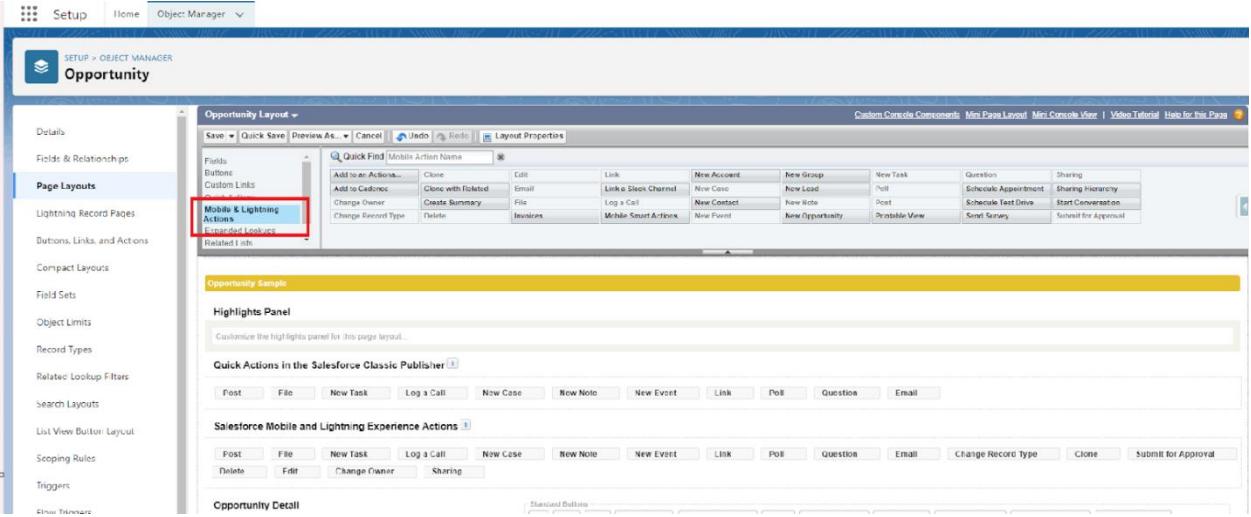


The screenshot shows the 'Page Layouts' section of the Opportunity Object Manager. The 'Page Layouts' link in the sidebar is highlighted with a red box. The main area displays a list of page layouts:

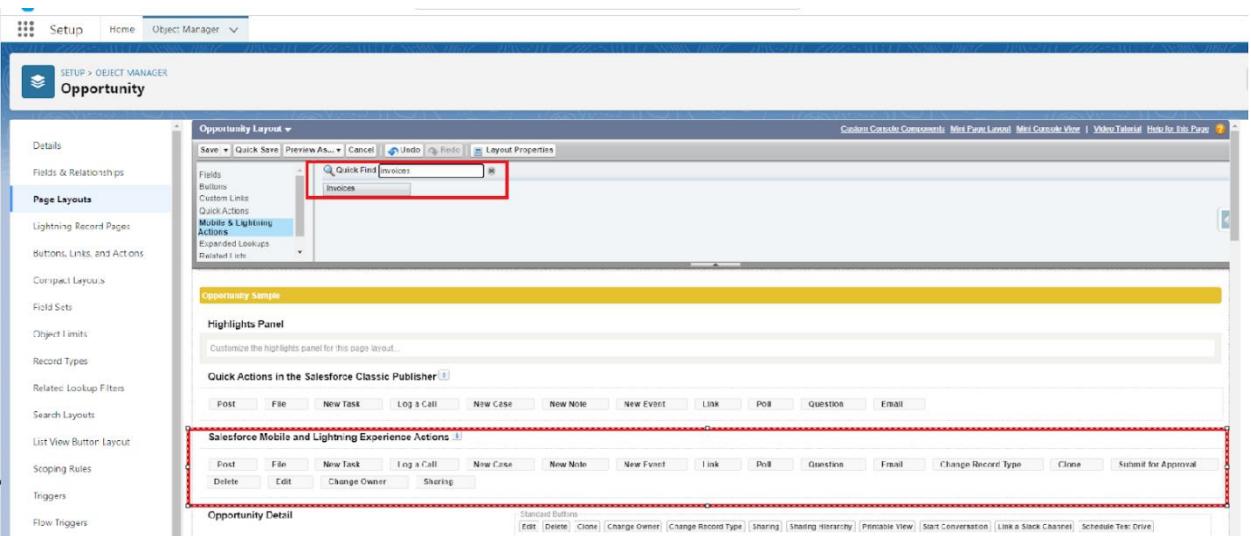
PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Opportunity (Marketing) Layout	Mohammad Sameer, 22/11/2023, 2:19 pm	Mohammad Sameer, 28/11/2023, 9:45 am
Opportunity (Sales) Layout	Mohammad Sameer, 22/11/2023, 2:19 pm	Mohammad Sameer, 28/11/2023, 9:45 am
Opportunity (Support) Layout	Mohammad Sameer, 22/11/2023, 2:19 pm	Mohammad Sameer, 28/11/2023, 9:45 am
Opportunity Layout	Mohammad Sameer, 22/11/2023, 2:19 pm	Mohammad Sameer, 01/12/2023, 10:57 am

Red boxes highlight the 'Page Layouts' link in the sidebar and the 'Opportunity Layout' row in the list.

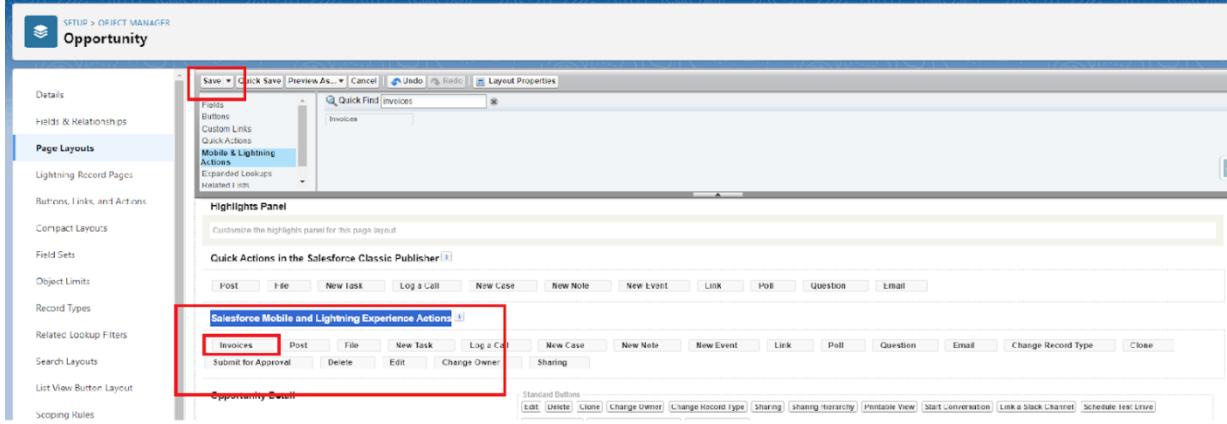
3. Click on Mobile And Lightning Action as show on below Image



4. Search for invoice on Quick Find.



5. Drag and Drop the Invoice into Salesforce Mobile and Lightning Experience Actions.



Click on Save.

## Step :29

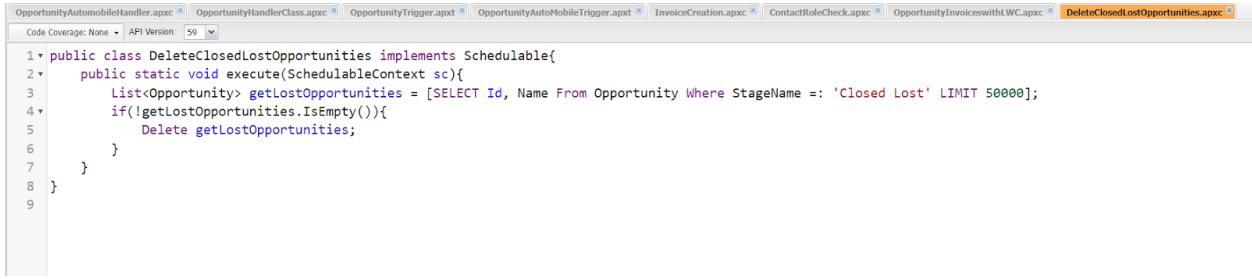
### Delete opportunity Schedule Class

#### Objective :

- Through this schedulable class, we can see all the Closed Lost Opportunities.
- We can delete all the Closed lost Opportunities by this Scheduled method on every monday as weekly.

1. Login to the respective 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 “DeleteClosedLostOpportunities ”

#### CODE SNIPPET :



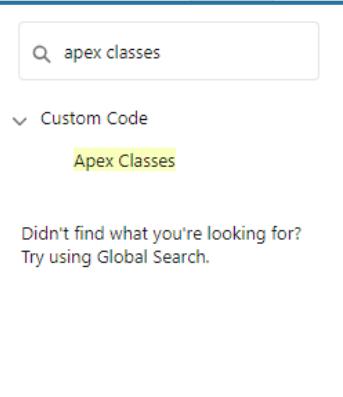
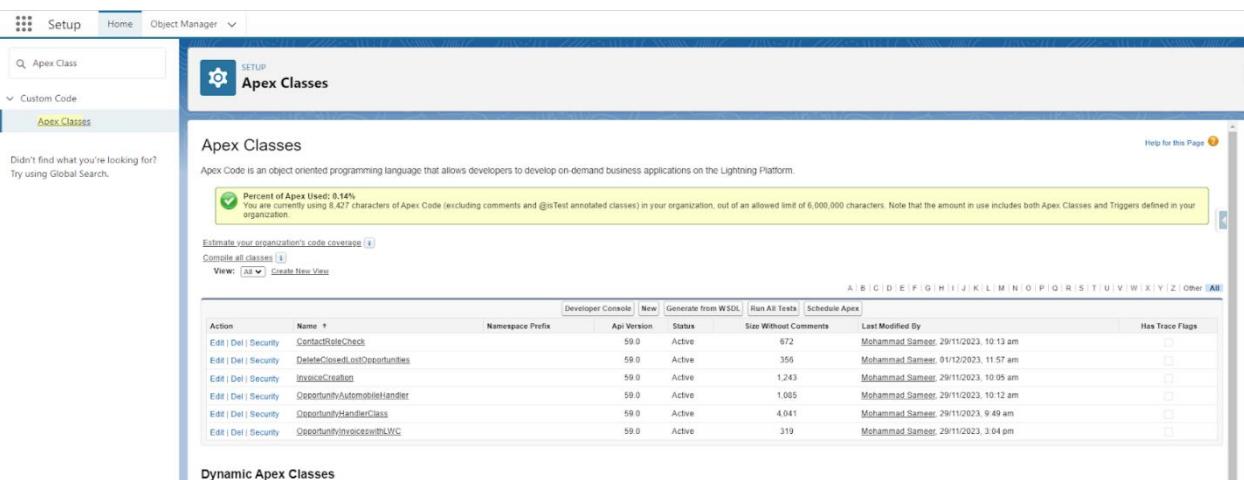
```
OpportunityAutomobileHandler.apxc OpportunityHandlerClass.apxc OpportunityTrigger.apxt OpportunityAutoMobileTrigger.apxt InvoiceCreation.apxc ContactRoleCheck.apxc OpportunityInvoiceswithWC.apxc DeleteClosedLostOpportunities.apxc
Code Coverage: None - API Version 59
1 public class DeleteClosedLostOpportunities implements Schedulable{
2     public static void execute(SchedulableContext sc){
3         List<Opportunity> getLostOpportunities = [SELECT Id, Name From Opportunity Where StageName =: 'Closed Lost' LIMIT 50000];
4         if(!getLostOpportunities.IsEmpty()){
5             Delete getLostOpportunities;
6         }
7     }
8 }
9
```

```
public class DeleteClosedLostOpportunities implements Schedulable{
    public static void execute(SchedulableContext sc){
        List<Opportunity> getLostOpportunities = [SELECT Id, Name From Opportunity Where StageName =: 'Closed Lost' LIMIT 50000];
        if(!getLostOpportunities.IsEmpty()){
            Delete getLostOpportunities;
        }
    }
}
```

## Schedule the Apex class:

- Go to the Home page in your salesforce account.

- In the search bar, enter Apex and click on Apex Classes.

Q Apex Class

Custom Code

Apex Classes

Didn't find what you're looking for?  
Try using Global Search.

Percent of Apex Used: 0.14%  
Currently using 6,427 characters of Apex Code (excluding comments and @isTest annotated classes) in your organization, out of an allowed limit of 6,000,000 characters. Note that the amount in use includes both Apex Classes and Triggers defined in your organization.

Estimate your organization's code coverage [\[i\]](#)

Compile all classes [\[i\]](#)

View: All [\[▼\]](#) Create New View

Action	Name <a href="#">↑</a>	Namespace Prefix	API Version	Status	Size Without Comments	Last Modified By	Has Trace Flags
Edit   Del   Security	ContactRoleCheck		59.0	Active	672	Mohammad Sameer	<input type="checkbox"/>
Edit   Del   Security	DeleteClosedlostOpportunities		59.0	Active	356	Mohammad Sameer	<input type="checkbox"/>
Edit   Del   Security	InvoiceCreation		59.0	Active	1,243	Mohammad Sameer	<input type="checkbox"/>
Edit   Del   Security	OpportunityAutomobileHandler		59.0	Active	1,085	Mohammad Sameer	<input type="checkbox"/>
Edit   Del   Security	OpportunityHandlerClass		59.0	Active	4,041	Mohammad Sameer	<input type="checkbox"/>
Edit   Del   Security	OpportunityInvoicingUtilWC		59.0	Active	319	Mohammad Sameer	<input type="checkbox"/>

Dynamic Apex Classes

- Click on Schedule Apex and enter the Job name.

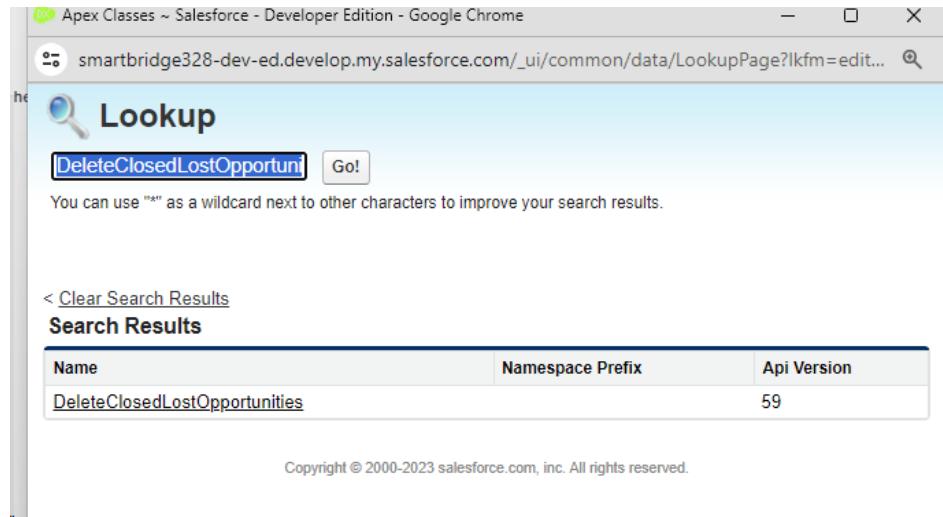
o Job Name : **DeleteOpportunitySchedule**

Schedule Apex

Schedule an Apex class that implements the 'Schedulable' interface to be automatically executed on a weekly or monthly interval.

Schedule Apex Execution	
Job Name	<input type="text" value="DeleteOpportunitySchedule"/>
Apex Class	<input type="text" value="DeleteClosedLostOpportuni"/> 
Frequency	<input checked="" type="radio"/> Weekly <input type="radio"/> Monthly
Start	<input type="text" value="01/12/2023"/> [ 01/12/2023 ]
End	<input type="text" value="01/01/2024"/> [ 01/12/2023 ]
Preferred Start Time	<input type="text" value="10:00 am"/> 
Exact start time will depend on job queue activity.	

- Now click on the search icon present near the Apex class : Goto the Lookup icon beside ? click on it ? select DeleteClosedLostOpportunities.



Apex Classes ~ Salesforce - Developer Edition - Google Chrome

smartbridge328-dev-ed.develop.my.salesforce.com/\_ui/common/data/LookupPage?lkfm=edit... 

## Lookup



You can use "\*" as a wildcard next to other characters to improve your search results.

< Clear Search Results

**Search Results**

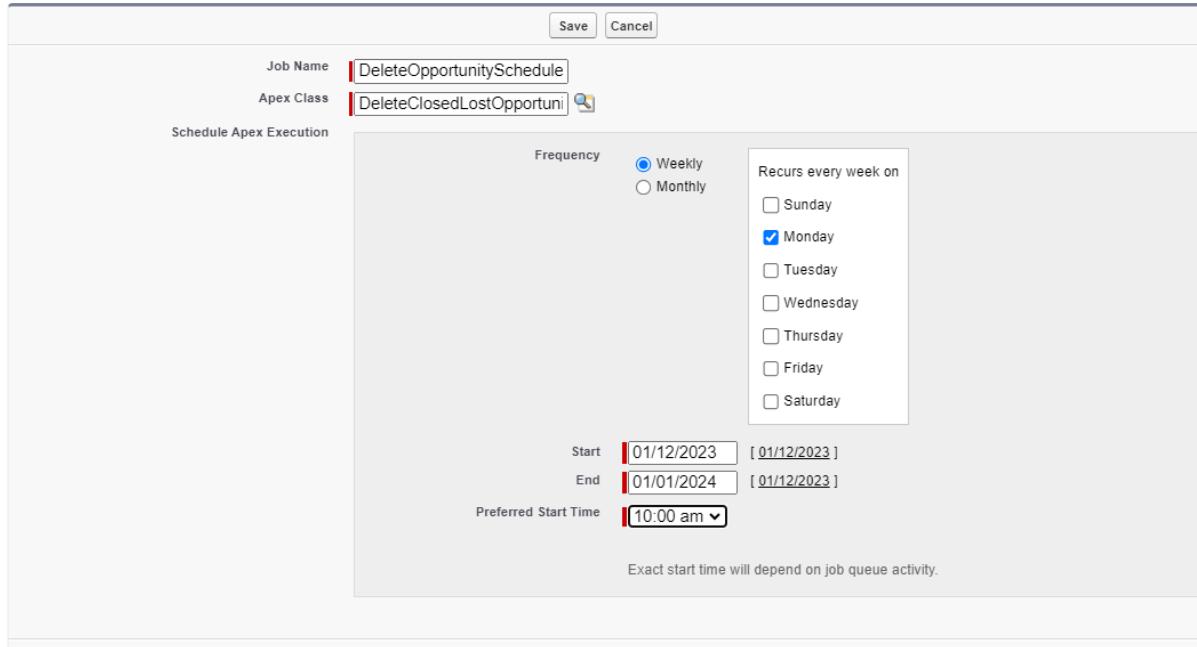
Name	Namespace Prefix	Api Version
<a href="#">DeleteClosedLostOpportunities</a>		59

Copyright © 2000-2023 salesforce.com, inc. All rights reserved.

- In the Schedule Apex section , select weekly and select Monday mentioned and preferred time as 10:00 AM.

## Schedule Apex

Schedule an Apex class that implements the 'Schedulable' interface to be automatically executed on a weekly or monthly interval.



Job Name: DeleteOpportunitySchedule

Apex Class: DeleteClosedLostOpportunit

Schedule Apex Execution

Frequency: Weekly (selected)

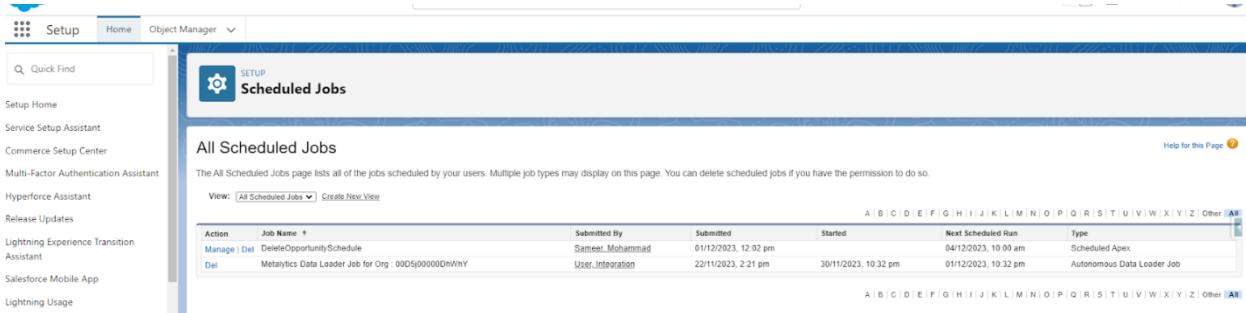
Start: 01/12/2023 [ 01/12/2023 ]

End: 01/01/2024 [ 01/12/2023 ]

Preferred Start Time: 10:00 am

Exact start time will depend on job queue activity.

- Click on Save. Now enter Apex in the search box and select Apex jobs.



All Scheduled Jobs

Action	Job Name	Submitted By	Submitted	Started	Next Scheduled Run	Type
Manage   Del	DeleteOpportunitySchedule	Sameer.Mohammad	01/12/2023, 12:02 pm		04/12/2023, 10:09 am	Scheduled Apex
Del	Analytics Data Loader Job for Org : 00D500000DwWY	User_Interopration	22/11/2023, 2:21 pm	30/11/2023, 10:32 pm	01/12/2023, 10:32 pm	Autonomous Data Loader Job

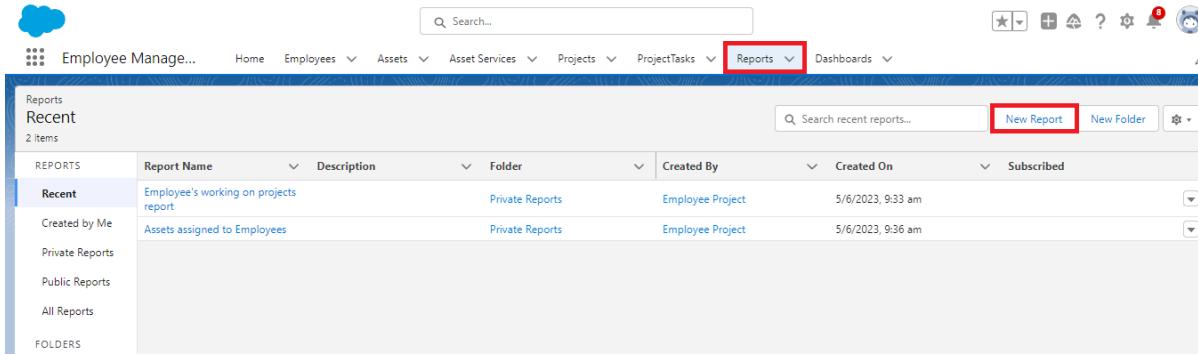
You can see that the batch job is in queue and will run whenever the day mentioned comes.

## Step :30

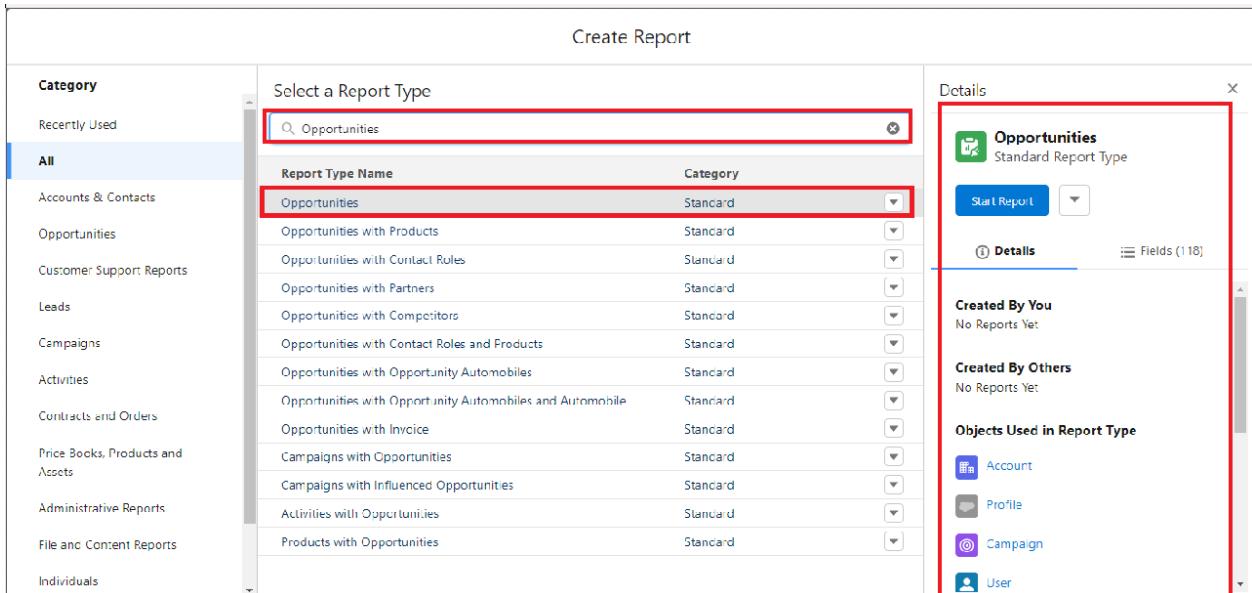
### Create Report on Opportunity

- Go to the app >> click on the reports tab

## 2. Click New Report.



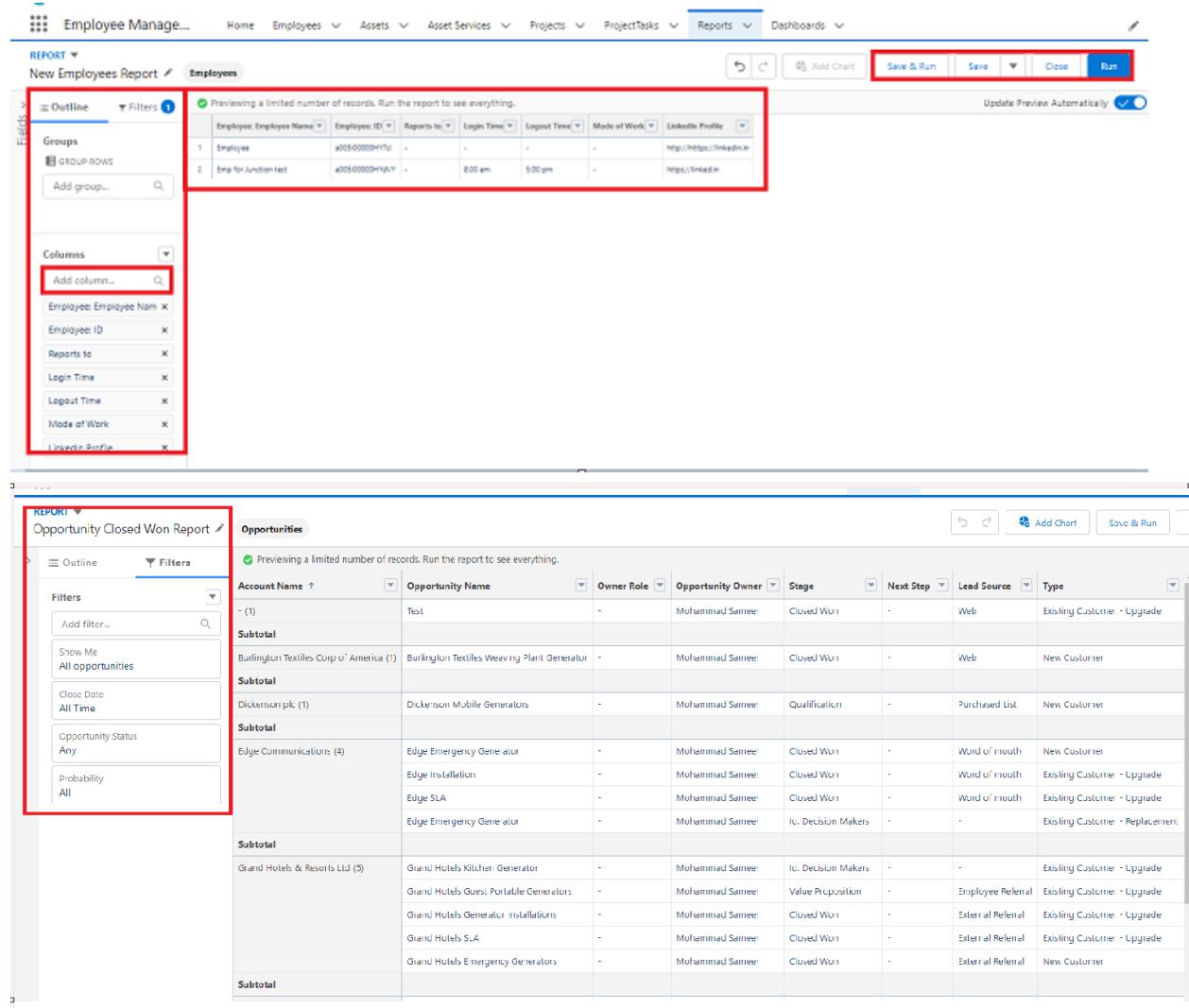
The screenshot shows the Salesforce Reports interface. At the top, there's a navigation bar with links like Home, Employees, Assets, Asset Services, Projects, ProjectTasks, Reports (which is currently selected and highlighted with a red box), and Dashboards. Below the navigation is a search bar labeled "Search...". The main area is titled "Reports Recent" and shows a list of recent reports. A red box highlights the "New Report" button at the top right of this list. On the left, there's a sidebar with categories: Reports (Recent, Created by Me, Private Reports, Public Reports, All Reports) and Folders.



The screenshot shows the "Create Report" dialog box. On the left, there's a sidebar with categories: Recently Used, 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 "Opportunities" category is selected and highlighted with a red box. In the center, there's a search bar labeled "Select a Report Type" with "Opportunities" typed in. Below it is a table with columns "Report Type Name" and "Category". The first row, "Opportunities", is highlighted with a red box. To the right, there's a "Details" panel for the "Opportunities" report type. This panel includes a "Start Report" button, a "Details" section showing "Fields (118)", and sections for "Created By You" and "Created By Others" (both showing "No Reports Yet"). At the bottom, there's a "Objects Used in Report Type" section listing Account, Profile, Campaign, and User.

## 2. Customize your report

- Add fields from left pane as shown below



The screenshot displays two reports within the Smart Internz application:

- New Employees Report:** This report shows a preview of employee data with two entries. The columns include Employee Name, Employee ID, Reports To, Login Time, Logout Time, Mode of Work, and LinkedIn Profile. A red box highlights the "Filters" section on the left, which contains fields for Employee Name, Employee ID, Reports To, Login Time, Logout Time, Mode of Work, and LinkedIn Profile. Another red box highlights the "Add column..." button in the "Columns" section.
- Opportunity Closed Won Report:** This report shows a preview of opportunities with various details like Opportunity Name, Owner Role, Opportunity Owner, Stage, Next Step, Lead Source, and Type. A red box highlights the "Filters" section on the left, which includes filters for Account Name, Opportunity Name, Owner Role, Opportunity Owner, Stage, Next Step, Lead Source, and Type. It also lists specific filters: "Show Me All opportunities", "Close Date All Time", "Opportunity Status Any", and "Probability All".

Add the Above Filter as well.

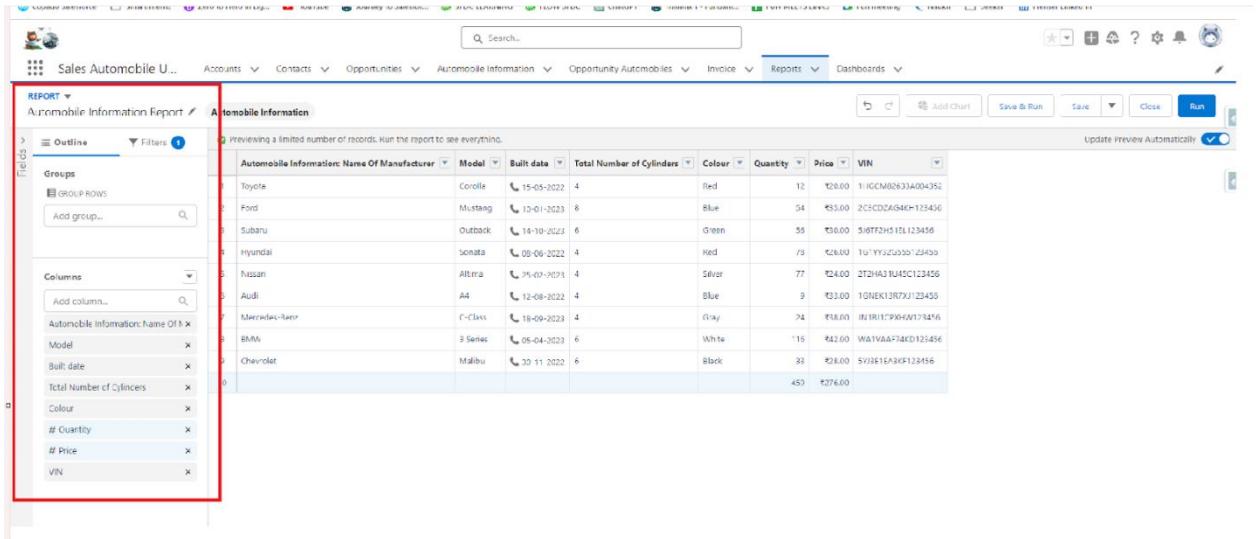
3. Save or run it.

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

## Step :31

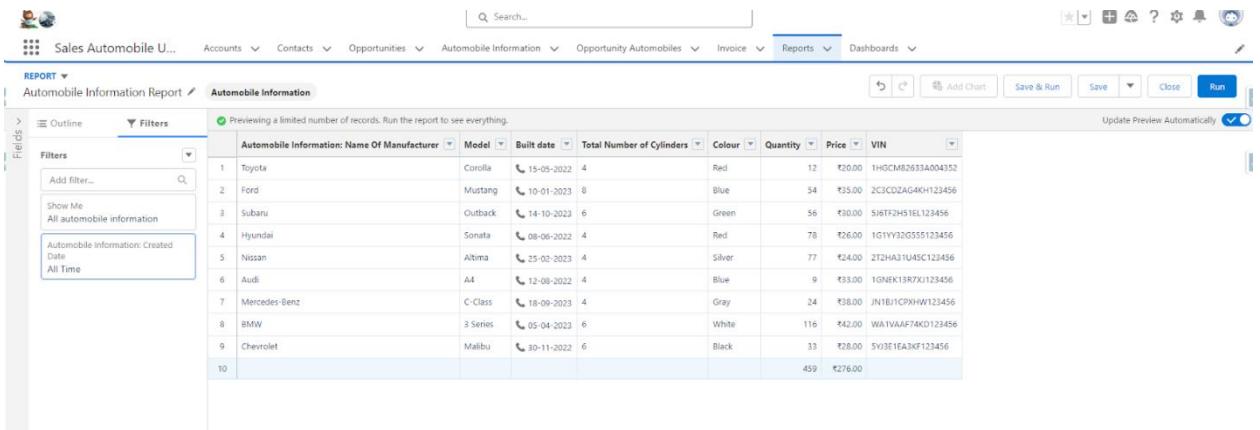
### Create Report on Automobile Information

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



The screenshot shows the 'Automobile Information' report in the Smartbridge application. The 'Fields' section on the left is highlighted with a red box. It lists various fields such as Name Of Manufacturer, Model, Built date, Total Number of Cylinders, Colour, Quantity, Price, and VIN. Under 'Filters', there is a dropdown set to 'All automobile information'. The main table displays 10 rows of automobile data.

	Name Of Manufacturer	Model	Built date	Total Number of Cylinders	Colour	Quantity	Price	VIN
1	Toyota	Corolla	15-05-2022	4	Red	12	\$20,000	1HGCM82633A004352
2	Ford	Mustang	10-01-2023	8	Blue	54	\$35,000	2C3CDZG6H4H123456
3	Subaru	Outback	14-10-2023	6	Green	56	\$30,000	5JFTH5H1E123456
4	Hyundai	Sonata	09-06-2022	4	Red	73	\$24,000	1UYY3C00000123456
5	Nissan	Altima	25-07-2023	4	Silver	77	\$24,000	2T2HAJ1M4C123456
6	Audi	A4	12-08-2022	4	Blue	9	\$31,000	1GNEK13R7UJ123456
7	Mercedes-Benz	C-Class	18-09-2023	4	Gray	24	\$38,000	JN1B1PXYH123456
8	BMW	3 Series	05-04-2023	6	White	116	\$42,000	WA1VAA74KD123456
9	Chevrolet	Malibu	30-11-2022	6	Black	33	\$28,000	SVJ3E15A2KF123456
10						459	\$276,000	



This screenshot shows the same 'Automobile Information' report interface. The 'Filters' section on the left is highlighted with a red box. It includes a dropdown set to 'All automobile information' and other filter options like 'Show Me' and 'Automobile Information: Created Date All Time'. The main table displays the same 10 rows of automobile data as the first screenshot.

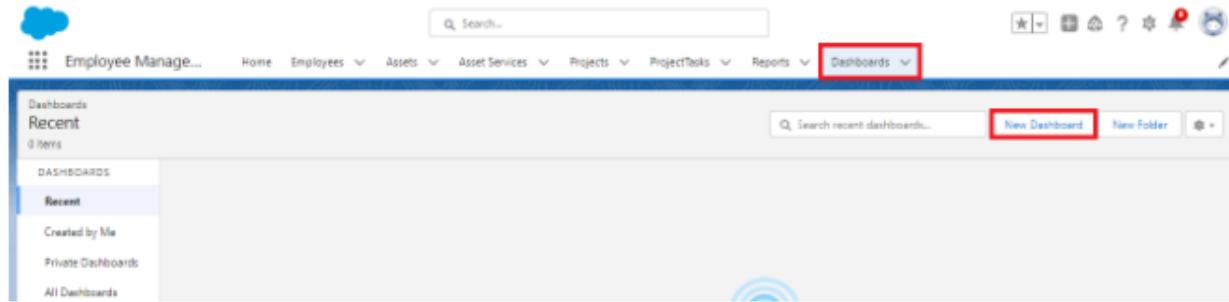
	Name Of Manufacturer	Model	Built date	Total Number of Cylinders	Colour	Quantity	Price	VIN
1	Toyota	Corolla	15-05-2022	4	Red	12	\$20,000	1HGCM82633A004352
2	Ford	Mustang	10-01-2023	8	Blue	54	\$35,000	2C3CDZG6H4H123456
3	Subaru	Outback	14-10-2023	6	Green	56	\$30,000	5JFTH5H1E123456
4	Hyundai	Sonata	09-06-2022	4	Red	73	\$24,000	1UYY3C00000123456
5	Nissan	Altima	25-07-2023	4	Silver	77	\$24,000	2T2HAJ1M4C123456
6	Audi	A4	12-08-2022	4	Blue	9	\$31,000	1GNEK13R7UJ123456
7	Mercedes-Benz	C-Class	18-09-2023	4	Gray	24	\$38,000	JN1B1PXYH123456
8	BMW	3 Series	05-04-2023	6	White	116	\$42,000	WA1VAA74KD123456
9	Chevrolet	Malibu	30-11-2022	6	Black	33	\$28,000	SVJ3E15A2KF123456
10						459	\$276,000	

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

## Step :31

Create Dashboard

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



2. Give a Name and click on Create.

### New Dashboard

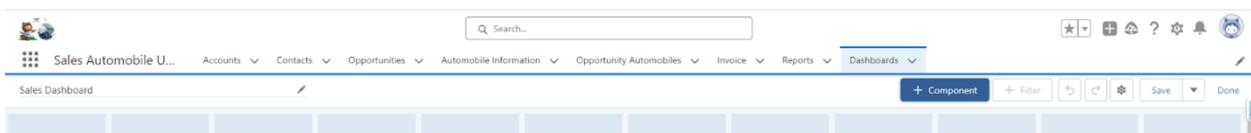
**\* Name**

**Description**

**Folder**

Name : Automobile Sales

3. Select add component.



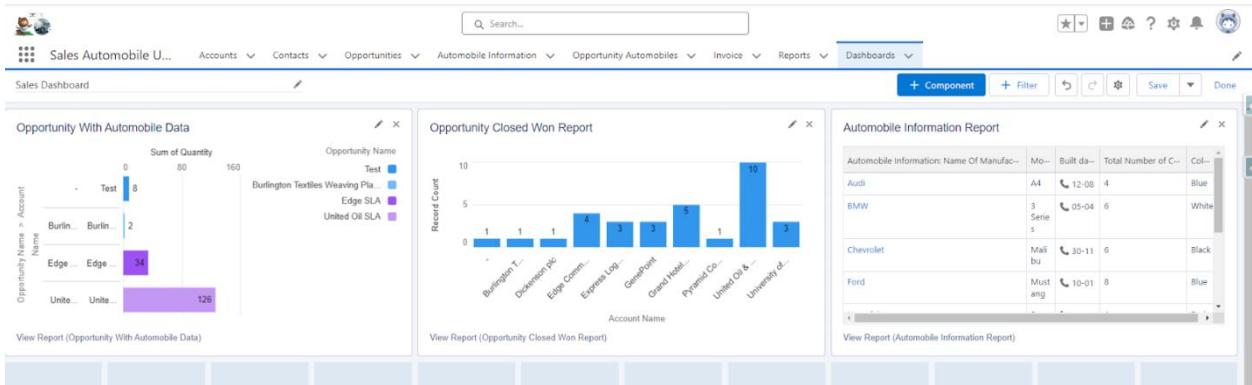
4. Select a Report and click on select.

Select Report

Reports	
<b>Recent</b>	
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
<b>Folders</b>	
Created by Me	Sample Flow Report: Screen Flows Automated Process - 22-Nov-2023, 2:19 pm - Public Reports
Shared with Me	
All Folders	

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

The Created Dashboard will look like this.



The dashboard displays the following components:

- Opportunity With Automobile Data:** A bar chart showing the sum of quantity for different opportunity names. The data is as follows:

Opportunity Name	Sum of Quantity
Test	8
Burlin...	2
Edge	34
Unite...	126

- Opportunity Closed Won Report:** A bar chart showing the record count for various account names. The data is as follows:

Account Name	Record Count
Burlington Textiles Weaving Pla...	1
Dickenson Inc.	1
Edge Comm...	4
Expense Log...	3
General Bus...	3
Grand Hotel...	5
Pyramid Co...	1
United Oil &...	19
University A...	3

- Automobile Information Report:** A table showing automobile information for different manufacturers. The data is as follows:

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

## 5. Testing and Validation

Describe the approach to testing:

- Unit Testing( Apex Classes ,Triggers).

User Interface Testing

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

This gives clarity that you are addressing various use cases or situations that Salesforce can handle during the implementation.

## 7. Conclusion

**Summary of Achievements :**[Brief summary of what has been accomplished]