



S.K.P ENGINEERING COLLEGE

Approved by AICTE & Affiliated to Anna University

NH 66, Somasipadi Post, Chinnakangiyar, Tiruvannamalai,

Tamil Nadu-606611

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BACHELOR OF ENGINEERING 2024-2025

FIFTH SEMESTER

PROJECT TITLE: SALES AUTOMOBIAL USING SALESFORCE(CRM)

Project Created by: Kannan. A, Kalanithi. B, Jagan Muthu. B, Hemajothi. S

Project Reviewed by: Mr.Nasrudeen sha (AP/CSE)

College Code: 5122

Team ID :



S.K.P ENGINEERING COLLEGE

Approved by AICTE & Affiliated to Anna University

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certified that this is a bonafide record of work done by,

Name : Kannan. A, Kalanithi. B, Jagan Muthu. B, Hemajothi. S

University Reg No :

Year / Semester : 3 / 5

Branch : CSE

Year : 2024-2025

Staff-in-Charge

Head of the Department

Submitted for the _____

Practical Examination held on_____

Internal Examiner

External Examiner

SALES AUTOMOBILE USING SALESFORCE

(CRM)

1. Project Overview:

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

2. Objectives:

- **Business Goals:**
 - ❖ Increase sales conversion rates.
 - ❖ Streamline customer data management and improve accessibility.
 - ❖ Enhance communication between sales teams and customers.
- **Specific Outcomes:**
 - ❖ Implementation of a centralized CRM system.
 - ❖ Development of automated workflows for lead management and follow-ups.
 - ❖ Generation of detailed sales reports for better decision-making.

3. Salesforce Key Features and Concepts Utilized :

- ❖ **Leads and Opportunities:** Efficient tracking and management of sales pipelines.
- ❖ **Automation Tools:** Use of workflows, process builders, and Lightning Flow to automate tasks like lead assignments and follow-ups.
- ❖ **Dashboards and Reports:** Custom dashboards to visualize sales performance and identify trends.
- ❖ **Integration:** Integration with third-party tools for enhanced functionality, such as email and marketing platforms.

- ❖ **Mobile Access:** Enable sales teams to access CRM features on-the-go using Salesforce's mobile app.

4. Detailed Steps to Solution Design:

- **Data Models:**
 - ❖ Created custom objects to track automobile inventory, customer preferences, and service history.
 - ❖ Established relationships between objects for comprehensive data analysis.
- **User Interface Design:**
 - ❖ Designed user-friendly Lightning pages for sales representatives to access critical information quickly.
 - ❖ Implemented dynamic forms to customize data views based on user roles.
- **Business Logic:**
 - ❖ Developed Apex triggers for automating complex business processes.
 - ❖ Defined validation rules to ensure data integrity.

Milestone 1-Salesforce :

Introduction:

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

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

What Is Salesforce?

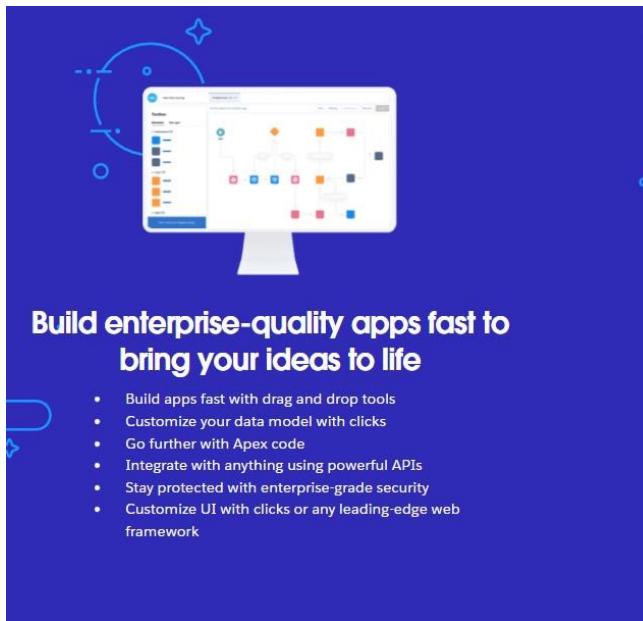
Salesforce is your customer success platform, designed to help you sell, service, market, analyze, and connect with your customers. Salesforce has everything you need to run your business from anywhere. Using standard products and features, you can manage relationships with prospects and customers, collaborate and engage with employees and partners, and store your data securely in the cloud. So what does that really mean? Well, before Salesforce, your contacts, emails, follow-up tasks, and prospective deals might have been organised something like this:

<https://youtu.be/r9EX3lGde5k>

Activity 1: Creating Developer Account:

Creating a developer org in salesforce.

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



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

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

Sign up for your Salesforce Developer Edition

A full-featured copy of the Platform, for free

Complete the form to start your free trial. Our team will be in touch to help you make the most of your trial.

First Name*	Last Name*
<input type="text" value="Your first name"/>	<input type="text" value="Your last name"/>
Email*	
<input type="text" value="Your email address"/>	
Role*	
<input type="text" value="Your job role"/>	
Company*	
<input type="text" value="Company Name"/>	

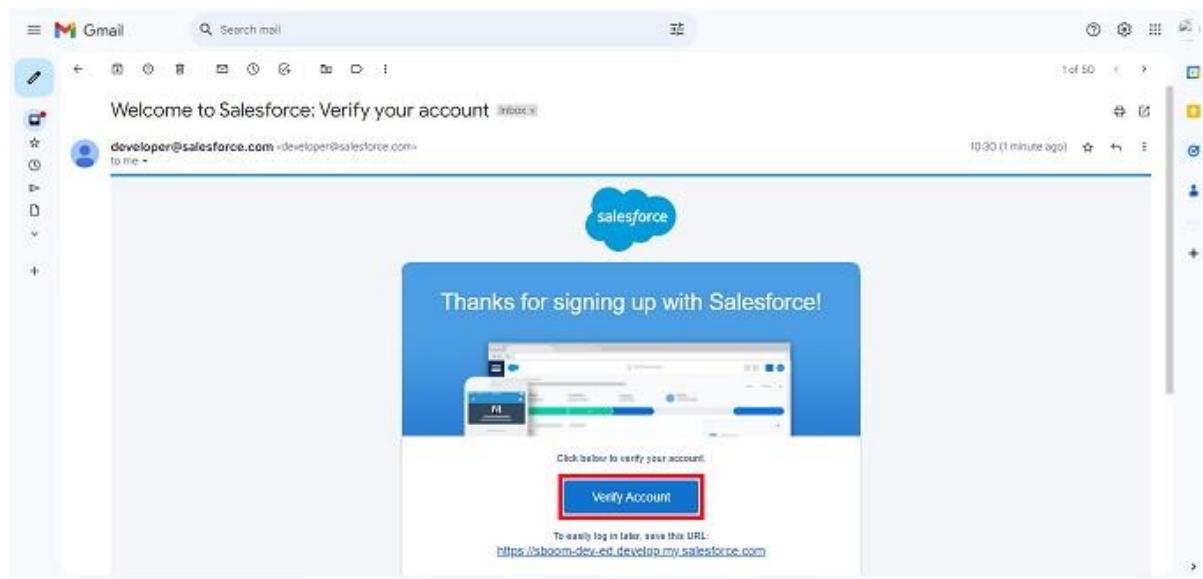
- 1) First name & Last name
- 2) Email
- 3) Role : Developer
- 4) Company : College Name
- 5) County : India
- 6) Postal Code : pin code
- 7) Username : should be a combination of your name and company

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.

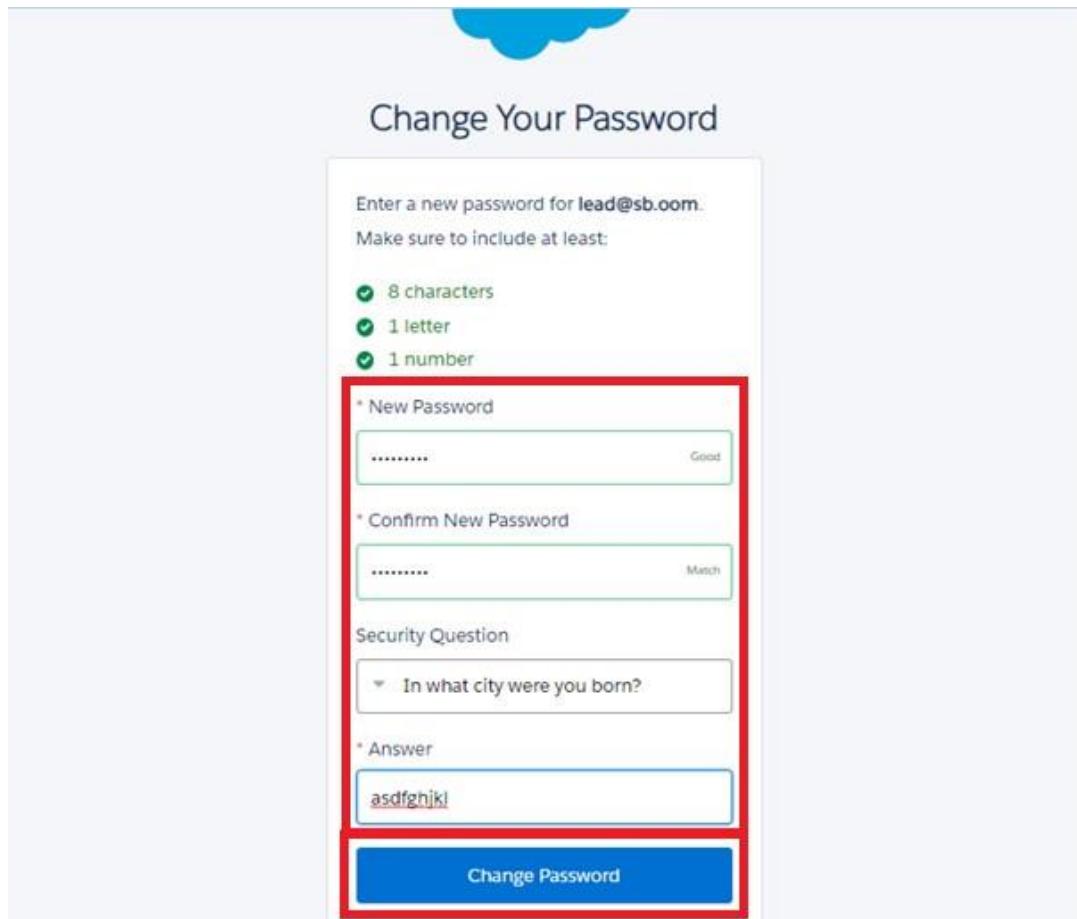
Activity 2:

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.

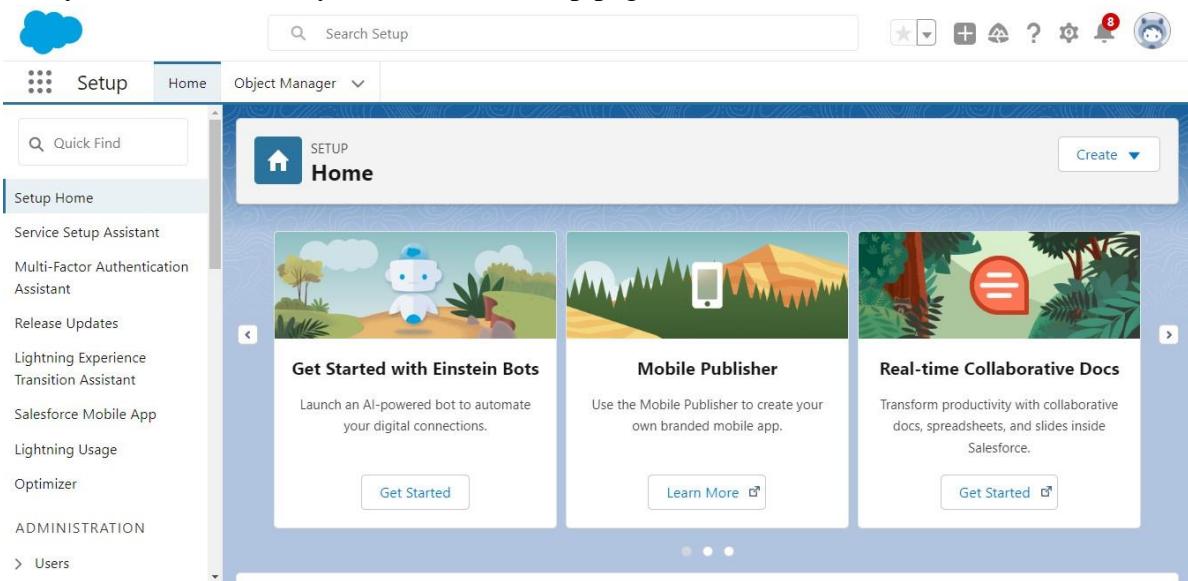


1. Click on Verify Account
2. Give a password and answer a security question and click on change password.



The screenshot shows the "Change Your Password" page for the email address lead@sb.com. The page instructs the user to enter a new password that includes at least 8 characters, 1 letter, and 1 number. It features two password input fields ("New Password" and "Confirm New Password"), both of which are highlighted with a red border. Below these are a "Security Question" dropdown set to "In what city were you born?" and an "Answer" input field containing "asdfghjkl". A large blue "Change Password" button at the bottom is also highlighted with a red border.

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



Milestone 2- Object

What Is an Object?

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

Salesforce objects are of two types:

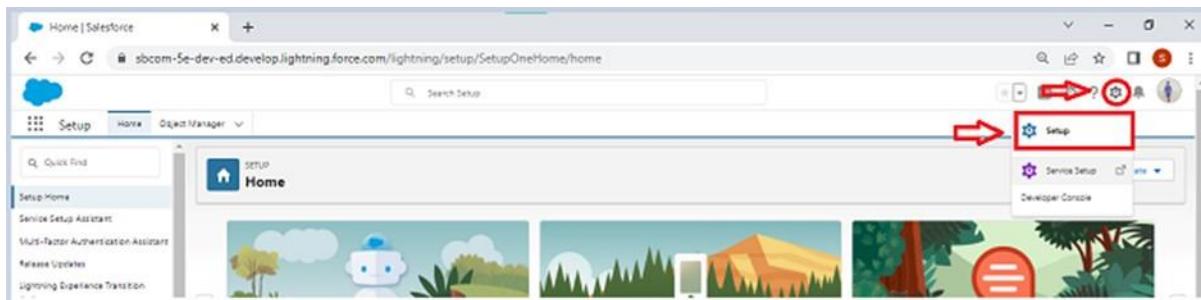
1. **Standard Objects:** Standard objects are the kind of objects that are provided by salesforce.com such as users, contracts, reports, dashboards, etc.
2. **Custom Objects:** Custom objects are those objects that are created by users. They supply information that is unique and essential to their organization.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

Use Case:

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

To Navigate to Setup page:

Click on gear icon → click setup.



To create an object:

Activity 1: 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.

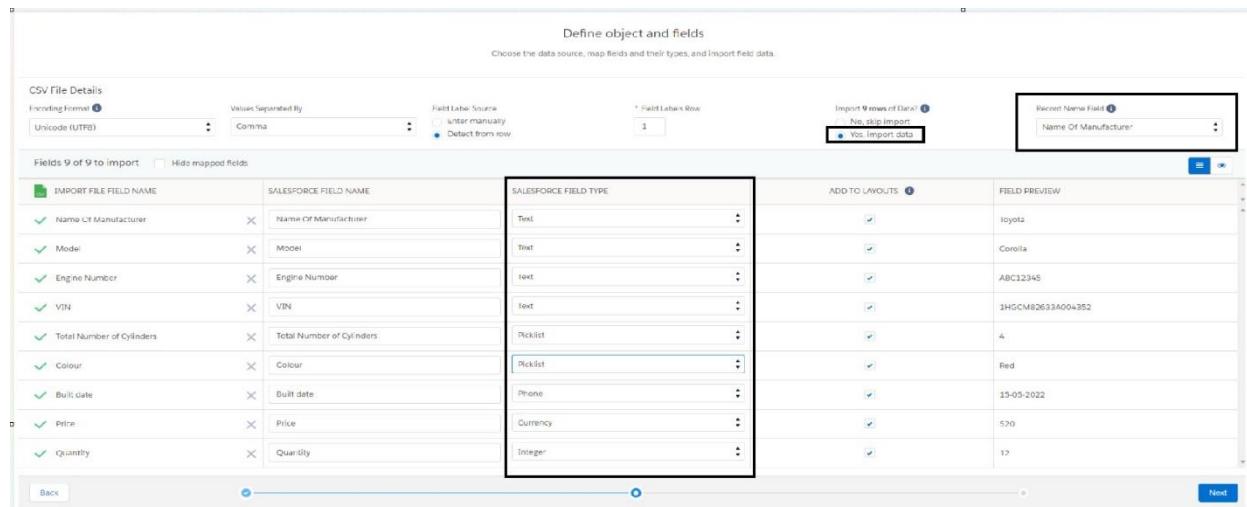
3. Click the Object Manager tab.



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.



The screenshot shows the 'Define object and fields' step of the Data Import Wizard. It displays a table mapping CSV fields to Salesforce fields. The 'SALESFORCE FIELD TYPE' column for the 'Name Of Manufacturer' field is highlighted with a black border. The table includes columns for 'IMPORT FILE FIELD NAME', 'SALESFORCE FIELD NAME', 'SALESFORCE FIELD TYPE', 'ADD TO LAYOUTS', and 'FIELD PREVIEW'. The 'Record Name Field' is set to 'Name Of Manufacturer'.

IMPORT FILE FIELD NAME	SALESFORCE FIELD NAME	SALESFORCE FIELD TYPE	ADD TO LAYOUTS	FIELD PREVIEW
Name Of Manufacturer	Name Of Manufacturer	Text	<input checked="" type="checkbox"/>	Toyota
Model	Model	Text	<input checked="" type="checkbox"/>	Corolla
Engine Number	Engine Number	Text	<input checked="" type="checkbox"/>	ABC12345
VIN	VIN	Text	<input checked="" type="checkbox"/>	1HGCM8D633A004352
Total Number of Cylinders	Total Number of Cylinders	Picklist	<input checked="" type="checkbox"/>	4
Colour	Colour	Picklist	<input checked="" type="checkbox"/>	Red
Built Date	Built date	Date	<input checked="" type="checkbox"/>	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.

Activity 2: 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

Encoding Format	Value Separated By	Field Label Source	Import 0 rows of Data
Unicode (UTF8)	Comma	Enter manually	No skip Import
		Detect from row	Yes, Import Later

Fields 5 of 5 to import Hide mapped fields

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

Activity 3 : Create Automobile Object

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

To create an object:

- From the setup page → Click on Object Manager → Click on Create → Click on Custom Object.

The screenshot shows the Salesforce Setup interface. At the top, there are tabs for 'Setup' and 'Object Manager'. The 'Object Manager' tab is highlighted with a red box. In the center, there's a search bar labeled 'Search Setup'. Below the search bar, there's a 'Create' button with a red box around it. On the right side of the screen, there's a 'Custom Object' button with a red box around it.

- Enter the label name → Opportunity Automobile
- Plural label name → Opportunity Automobiles

The screenshot shows the 'New Custom Object' page in the Salesforce Setup. The title bar says 'SETUP New Custom Object'. The main section is titled 'Custom Object Information'. It has fields for 'Label' (with an example 'Account') and 'Plural Label' (with an example 'Accounts'). There's also a checkbox for 'Starts with vowel sound'. Below this, there's a field for 'Object Name' (with an example 'Account') and a 'Description' text area. Under 'Context-Sensitive Help Setting', there are two radio buttons: 'Open the standard Salesforce.com Help & Training window' (selected) and 'Open a window using a Visualforce page'. A 'Content Name' dropdown is set to 'None'. At the bottom, there's a section titled 'Enter Record Name Label and Format' with a note about record names appearing in various contexts. It has a 'Record Name' field (example 'Account Name') and a 'Data Type' dropdown set to '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

—None—

Enter Record Name Label and Format

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

Record Name	<input type="text" value="Opportunity Automobile Id"/>	Example: Account Name
Data Type	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 sears 4. → Save

Milestone 3 - Tabs

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

Types of Tabs:

1. Custom Tabs

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

2. Web Tabs

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

3. Visualforce Tabs

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

4. Lightning Component Tabs

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

5. Lightning Page Tabs

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

Use Case:

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

Activity 1: Creating a Custom Tab

To create a Tab: (Opportunity Automobile)

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

Custom Tabs

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

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



Custom Object Tabs

New What Is This?

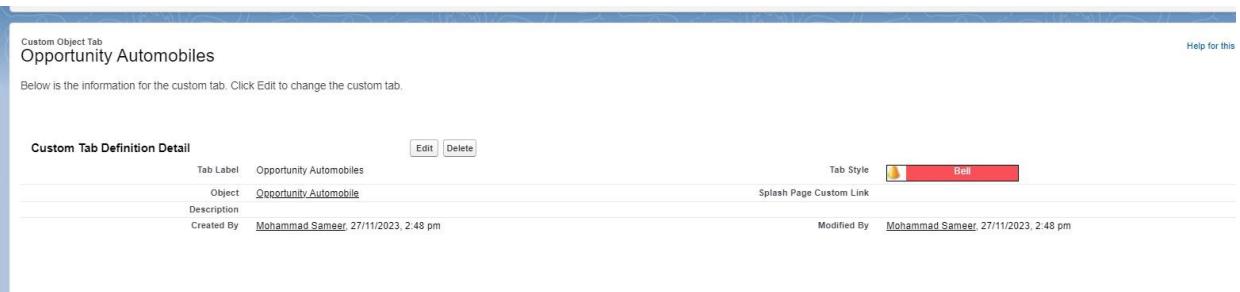
No Custom Object Tabs have been defined

Web Tabs

New What Is This?

No Web Tabs have been defined

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 Object Tab
Opportunity Automobiles

Below is the information for the custom tab. Click Edit to change the custom tab.

Custom Tab Definition Detail

Tab Label	Opportunity Automobiles	Edit	Delete
Object	Opportunity Automobile	Tab Style	Bell
Description		Splash Page Custom Link	
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.

Milestone 4- The Lightning App:

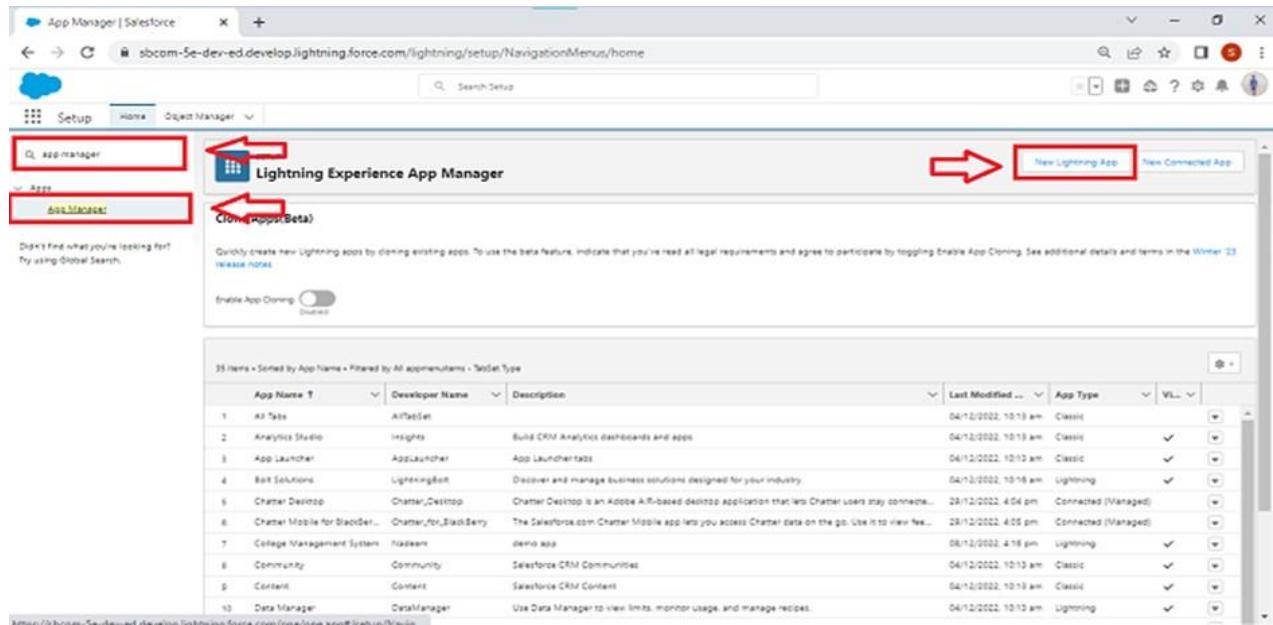
An app is a collection of items that work together to serve a particular function. In Lightning Experience, Lightning apps gives users access to sets of objects, tabs, and other items all in one convenient bundle in the navigation bar. Lightning apps let you brand your apps with a custom color and logo.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

Use Case:

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

Activity 1: Create a Lightning App

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

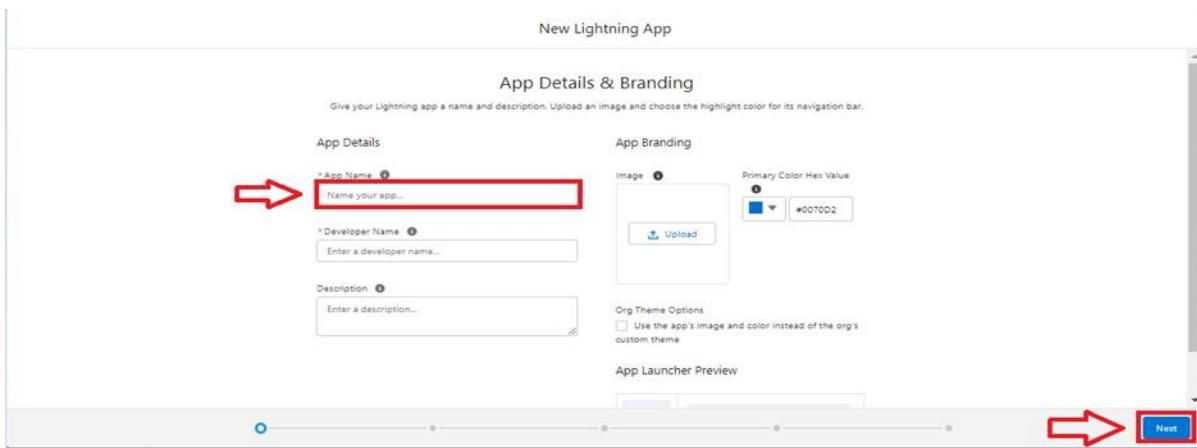


The screenshot shows the Salesforce App Manager interface. At the top, there are several buttons: 'App Manager', 'Lightning Experience App Manager' (which is highlighted with a red arrow), 'Clone (Beta)' (also highlighted with a red arrow), and 'New Connected App'. Below these buttons, there is a section titled 'Quickly create new Lightning apps by cloning existing apps. To use the beta feature, indicate that you've read all legal requirements and agree to participate by toggling Enable App Cloning. See additional details and terms in the Winter '23 Release Notes.' A toggle switch labeled 'Enable App Cloning' is set to 'Disabled'. The main area displays a table of existing apps, with the first few rows shown below:

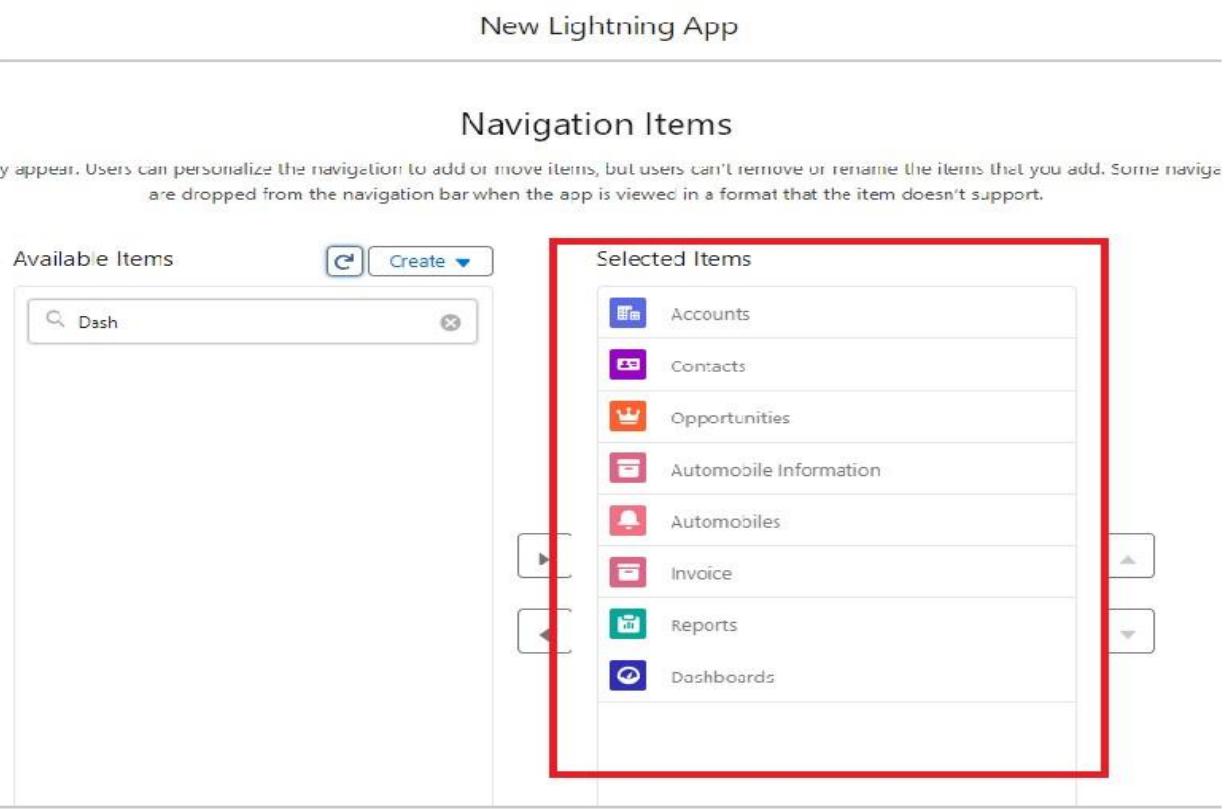
App Name	Developer Name	Description	Last Modified	App Type
All Tabs	AllTabSet	Build CRM Analytics dashboards and apps	04/12/2022, 10:19 am	Classic
Analytics Studio	Insights	Build CRM Analytics dashboards and apps	04/12/2022, 10:19 am	Classic
App Launcher	AppLauncher	App Launcher tabs	04/12/2022, 10:19 am	Classic
Bolt Solutions	LightningBolt	Discover and manage business solutions designed for your industry.	04/12/2022, 10:19 am	Lightning
Chatter Desktop	Chatter/Desktop	Chatter Desktop is an Adobe AIR-based desktop application that lets Chatter users stay connected...	28/12/2022, 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 feed...	28/12/2022, 4:05 pm	Connected (Managed)
College Management System	Nadeem	demo app	08/12/2022, 4:18 pm	Lightning
Community	Community	Salesforce CRM Communities	04/12/2022, 10:19 am	Classic
Content	Content	Salesforce CRM Content	04/12/2022, 10:19 am	Classic
Data Manager	DataManager	Use Data Manager to view limits, monitor usage, and manage recipes.	04/12/2022, 10:19 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
3. Then click Next → (App option page) keep it as default → Next → (Utility Items) keep it as default → Next.



4. To Add Navigation Items:

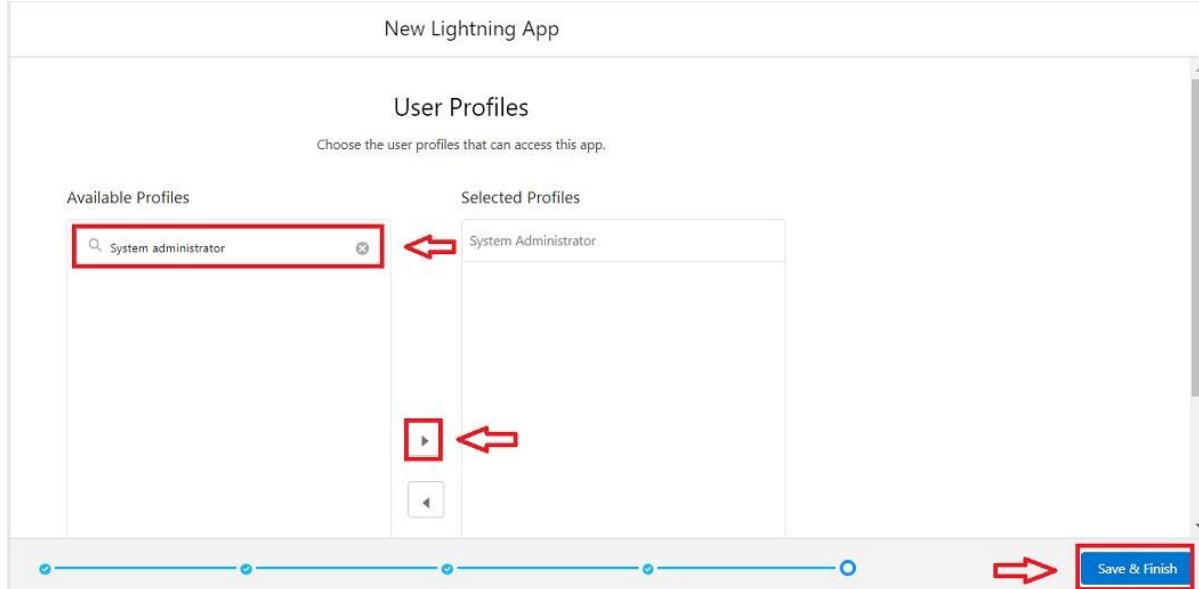


The screenshot shows the 'Navigation Items' configuration interface. On the left, there is a list titled 'Available Items' with a search bar containing 'Dash'. On the right, there is a list titled 'Selected Items' enclosed in a red border. The 'Selected Items' list contains the following items:

- Accounts
- Contacts
- Opportunities
- Automobile Information
- Automobiles
- Invoice
- Reports
- Dashboards

5. Search the items in the search bar(Account,Contact ,Opportunities,Automobile Information,Opportunity Automobile,Invoice, Reports, Dashboard) from the search bar and move it using the arrow button → Next.

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



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

Milestone 5:Fields & Relationships

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

Types of Fields

1. Standard Fields
2. Custom Fields

Standard Fields:

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

- Created By
- Owner
- Last Modified
- Field Made During object Creation

Custom Fields:

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

Use Case:

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

Activity 1: Creating Opportunity Master Detail Relationship Field in Opportunity AutoMobile Object

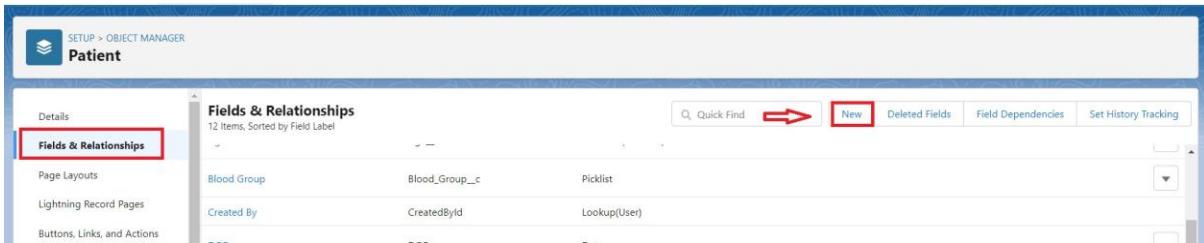
To create fields in an object:

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



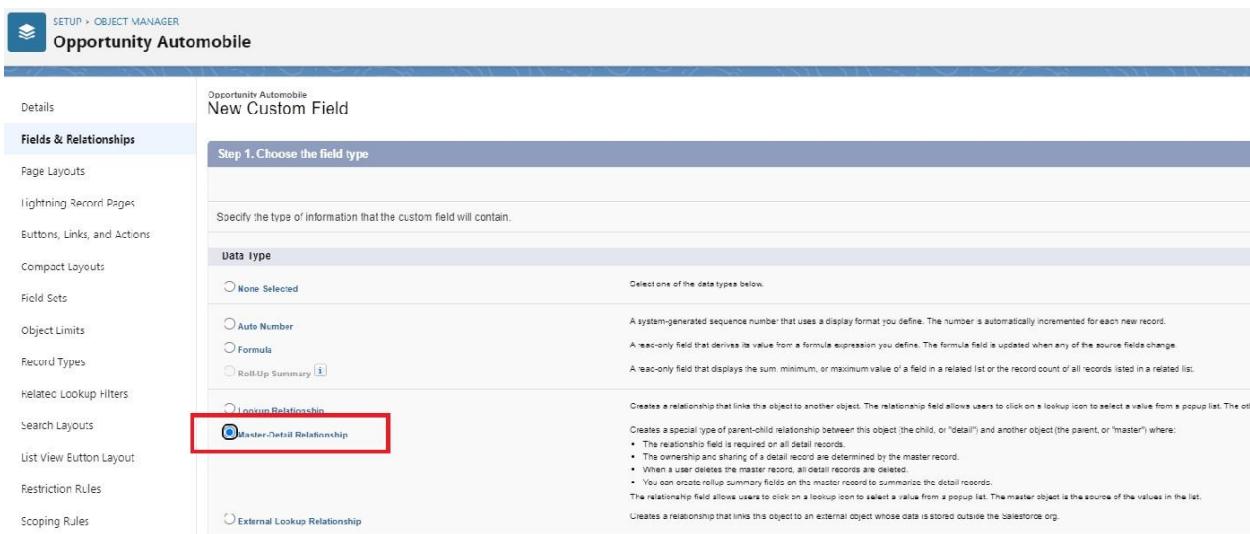
The screenshot shows the Salesforce Object Manager page. The search bar at the top contains "Opportunity Automobile". A red box highlights the search term. Below the search bar, the "Object Manager" section displays one item: "Opportunity Automobile" with the API name "Opportunity_Automobile__c" and Type "Custom Object". The "Last Modified" field shows "27/11/2023". The "DEPLOYED" status is checked. The "Create" button is visible on the right.

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



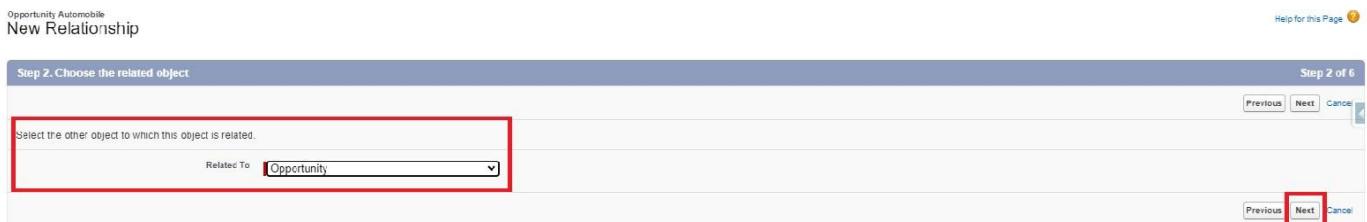
The screenshot shows the "Fields & Relationships" page for the "Patient" object. The "Fields & Relationships" tab is selected. A red box highlights the "New" button. Other tabs include "Details", "Page Layouts", "Lightning Record Pages", and "Buttons, Links, and Actions". The table lists fields like "Blood Group" and "Created By". The "New" button is highlighted with a red box.

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



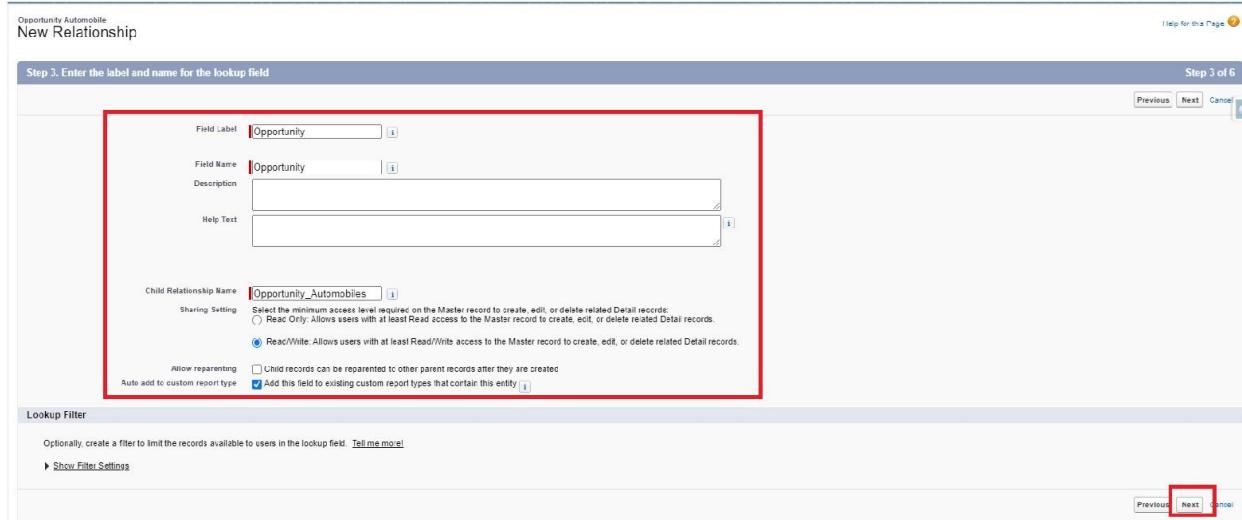
The screenshot shows the "New Custom Field" configuration page for the "Opportunity Automobile" object. The "Fields & Relationships" tab is selected. The "Data type" section shows the "Master-Detail Relationship" option selected, highlighted with a red box. Other options like "None Selected", "Auto Number", "Formula", "Rollup Summary", and "Lookup Relationship" are also listed.

4. Click on Next



The screenshot shows "Step 2 of 6: Choose the related object" for the new relationship. The "Related To" dropdown menu is open, showing "Opportunity" as the selected option, which is highlighted with a red box. The "Next" button at the bottom right is also highlighted with a red box.

5. Fill the above as following:
- Field Label: gets auto Generated(Opportunity) ○
 - Field Name : gets auto generated(Opportunity) ○
 - Click on Next → Next → Save and new.



Opportunity Automobile
New Relationship

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

Field Label: Opportunity

Field Name: Opportunity

Description:

Help Text:

Child Relationship Name: Opportunity_Automobiles

Sharing Setting: Select the minimum access level required on the Master record to create, edit, or delete related Detail records:
 Read Only: Allows users with at least Read access to the Master record to create, edit, or delete related Detail records.
 Read/Write: Allows users with at least Read/Write access to the Master record to create, edit, or delete related Detail records.

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

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

Lookup Filter

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

Show Filter Settings

Step 3 of 6

Previous Next Cancel

Activity2: Creating the AutoMobile Information Lookup Fieldin 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.



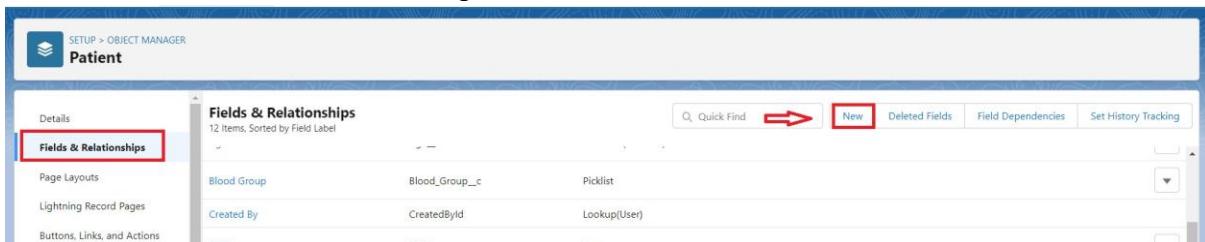
SETUP

Object Manager

1 Items, Sorted by Label

LABEL	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Opportunity Automobile	Opportunity_Automobile_L	Custom Object		27/11/2023	

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



SETUP > OBJECT MANAGER

Patient

Fields & Relationships

12 Items, Sorted by Field Label

Details	Fields & Relationships	Quick Find	New	Deleted Fields	Field Dependencies	Set History Tracking
Page Layouts	Blood Group	Blood_Group__c	Picklist			
Lightning Record Pages	Created By	CreatedById	Lookup(User)			
Buttons, Links, and Actions						

3. Select Data type as “Lookup RelationShip”.

SETUP > OBJECT MANAGER
Opportunity Automobile

Details

Fields & Relationships

- Page Layouts
- Lightning Record Pages
- Buttons, Links, and Actions
- Compact Layouts
- Field Sets
- Object Limits
- Record Types
- Related Lookup Filters

Data Type

None Selected

Auto Number

Formula

Roll-up Summary

Lookup Relationship

Master-Detail Relationship

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

Select one of the data types below.

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

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

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

Creates a relationship that links this object to another object. The relationship field allows users to click on a lookup icon to select a value from a popup list. This object is the child object in the relationship.

The relationship field is required on all detail records.

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

The relationship field allows users to click on a lookup icon to select a value from a popup list. The master object is the source of the values in the list.

4. Click on Next

Opportunity Automobile
New Relationship

Step 2. Choose the related object

Select the other object to which this object is related

Related To:

Help for this Page

Step

Previous Next Cancel

Previous Next Cancel

5. Fill the above as following:
 - a. Field Label: Automobile
 - b. Field Name : Automobile
6. Click on Next → Next → Save and new.

Opportunity Automobile
New Relationship

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

Step 3 of 6

Field Label:

Field Name:

Description:

Help Text:

Child Relationship Name:

Required: Always require a value in this field in order to save a record
 Clear the value of this field. You can't choose this option if you make this field required.
 Don't allow deletion of the lookup record that's part of a lookup relationship.

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

Lookup Filter:

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

Show Filter Settings

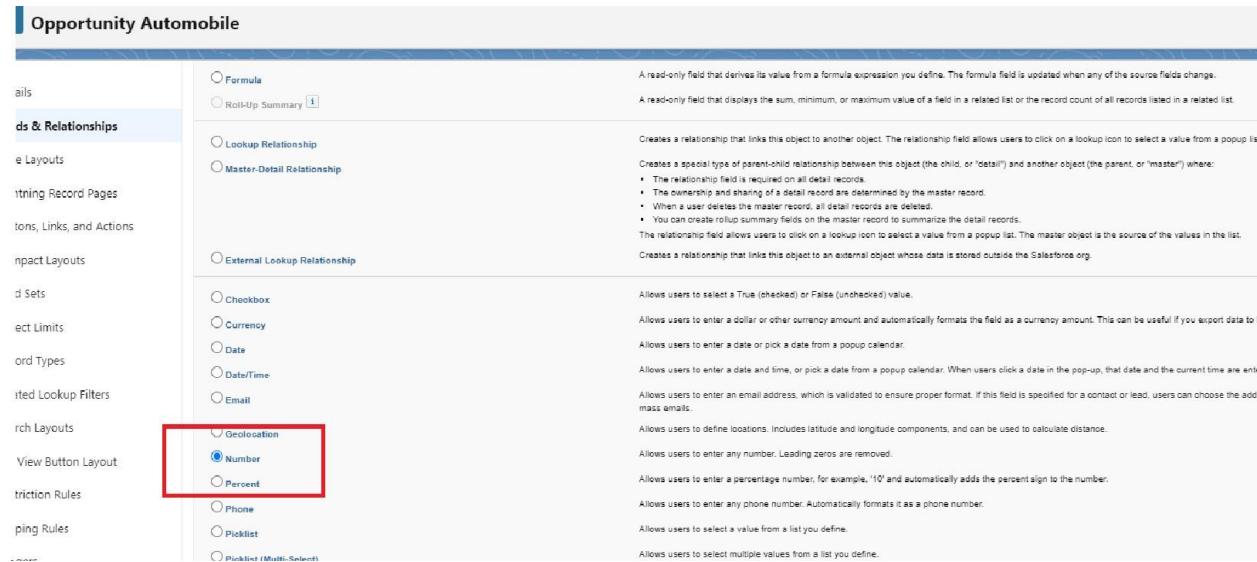
Previous Next Cancel

Activity 3: Creating Quantity Number Field in Opportunity Automobile Object

To create fields in an object:

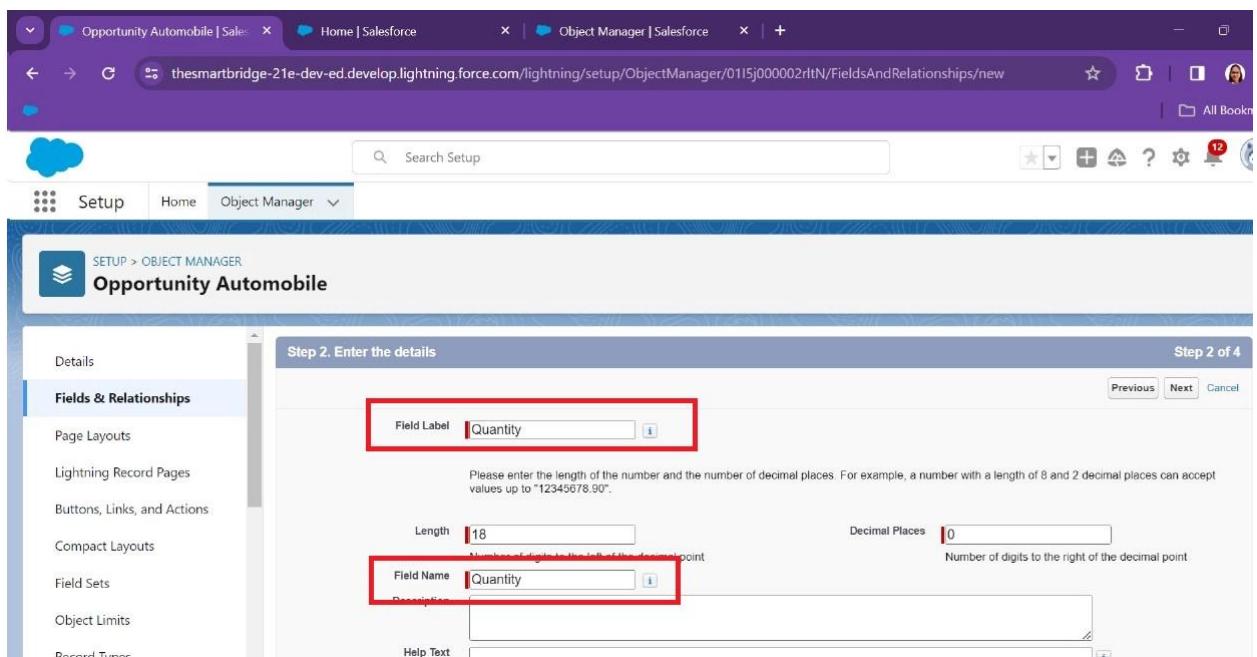
1. Go to setup → click on Object Manager → type object name(Opportunity Automobile) in quick find bar→ click on the object.
2. Now click on “Fields & Relationships” → New.

3. Select Data type as “Number” and click Next.



The screenshot shows the 'Field Types' section of the Salesforce documentation. On the left, there's a sidebar with links like 'Formula', 'Roll-up Summary', 'Lookup Relationship', 'Master-Detail Relationship', 'External Lookup Relationship', 'Checkbox', 'Currency', 'Date', 'Date/Time', 'Email', 'Geolocation', 'Number' (which is highlighted with a red box), 'Percent', 'Phone', 'Picklist', and 'Picklist (Multi-Select)'. To the right of each field type is a brief description and some bullet points or examples.

- a. Field Label → Quantity
- b. Field Name→ Quantity



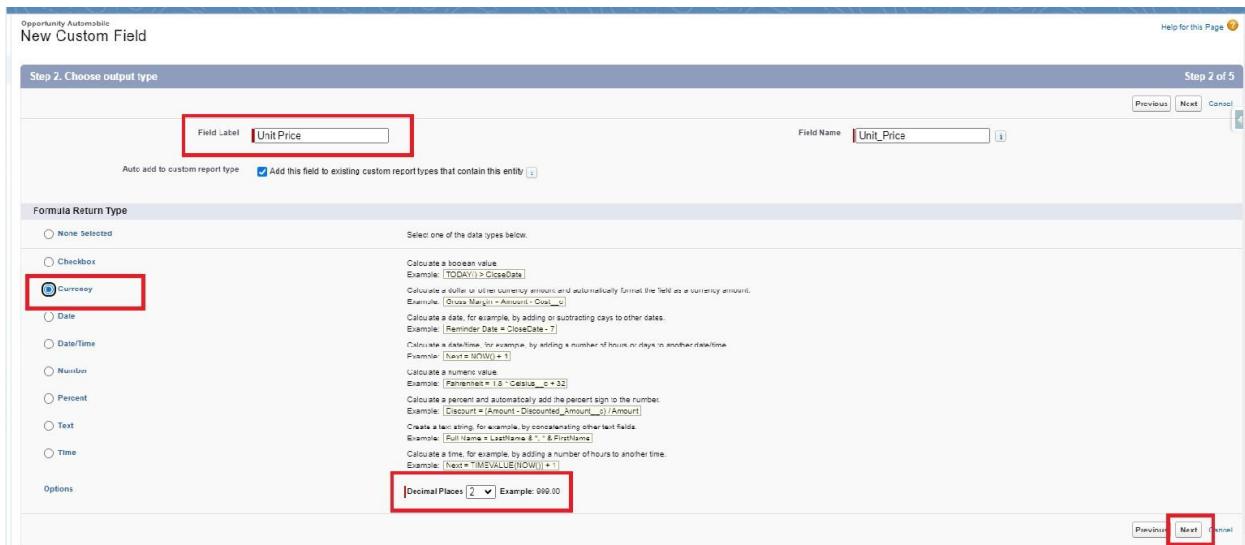
The screenshot shows the 'Object Manager' setup screen for creating a new field. The 'Field Label' is set to 'Quantity' (highlighted with a red box). The 'Length' is set to 18 and the 'Decimal Places' is set to 0. The 'Field Name' is also set to 'Quantity' (highlighted with a red box). Other fields like 'Description' and 'Help Text' are empty. The status bar at the top right says 'Step 2 of 4'.

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

Activity 4 : Creating Formula Field in Opportunity Automobile Object

To create fields in an object:

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



Opportunity Automobile
New Custom Field

Step 2. Choose output type

Field Label **Unit Price**

Field Name **Unit_Price**

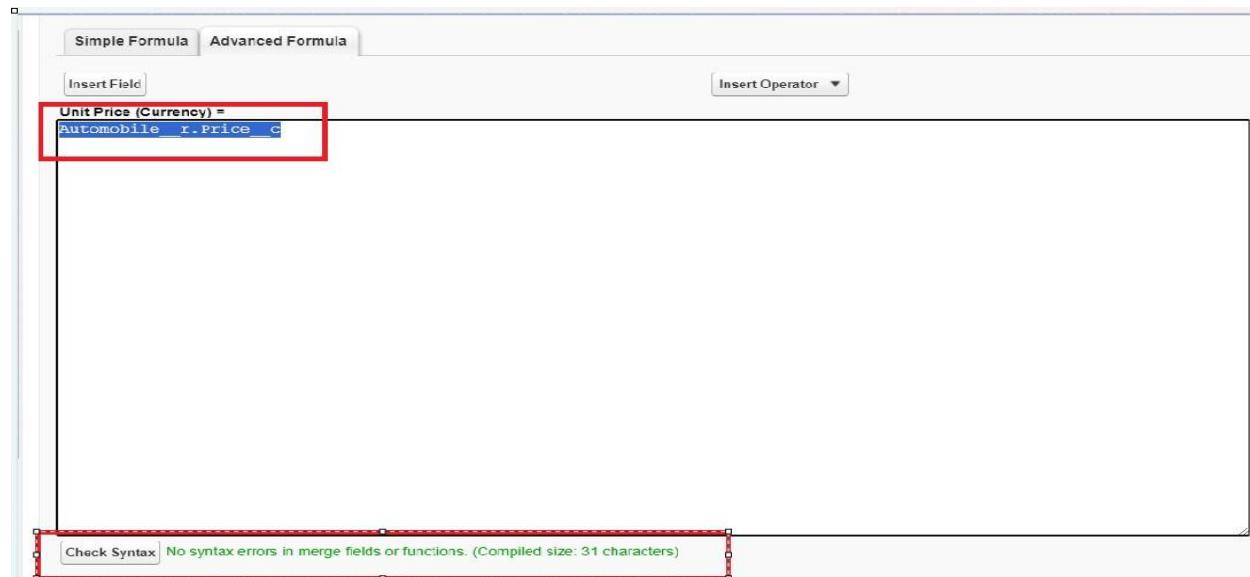
Formula Return Type

Currency

DecimalPlaces **2**

Next

4. Under Advanced Formula write down the formula : **Automobile__r.Price__c**



Simple Formula Advanced Formula

Insert Field Insert Operator

Unit Price (Currency) =
Automobile__r.Price__c

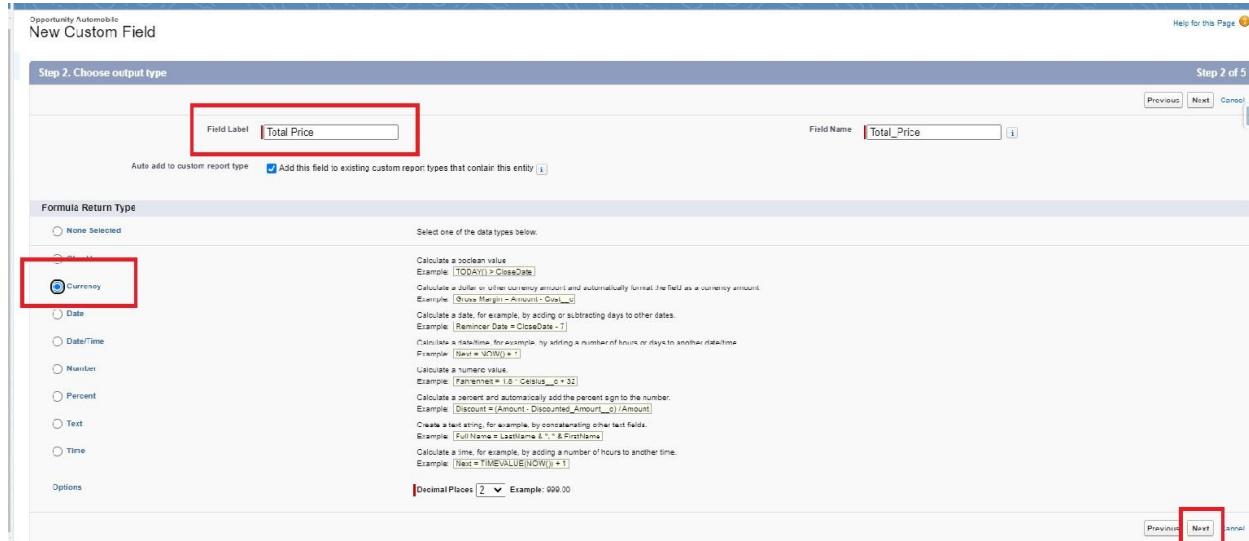
Check Syntax No syntax errors in merge fields or functions. (Compiled size: 31 characters)

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

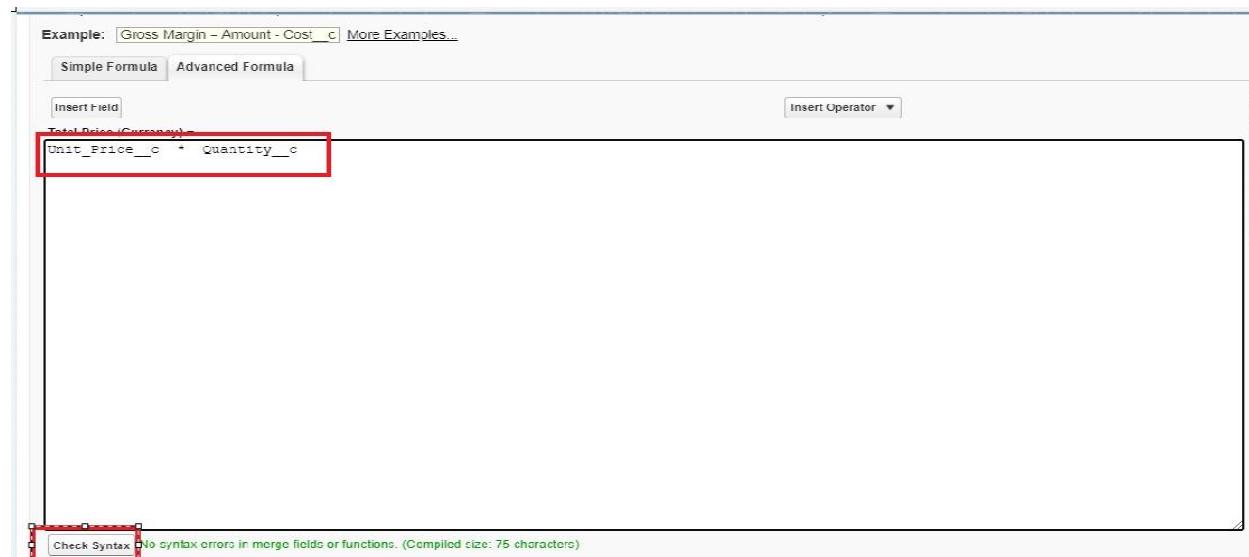
Activity 5 : Creating the Formula field in Opportunity Automobile Object

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.
2. Now click on “Fields & Relationships” → New.
3. Select Data type as “Formula” and click Next.
4. 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.



5.Under Advanced Formula write down the formula : **Unit_Price__c * Quantity__c**

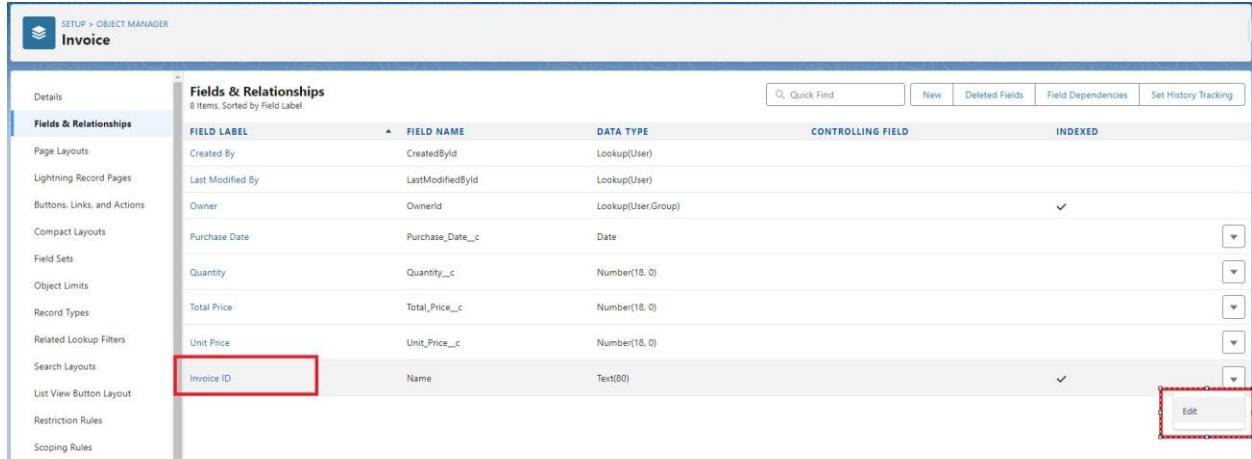


6. click “Check Syntax” and Next → Next→ Save.

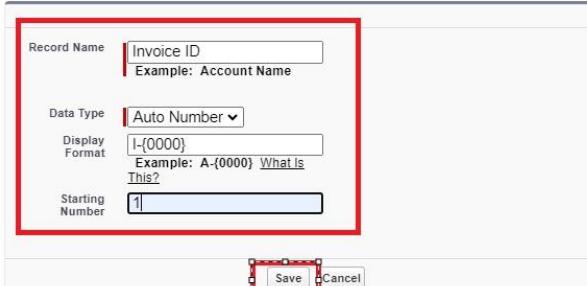
Activity 6 : Updating field in Invoice Object

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.



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



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

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

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

Activity 7 : 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>Field Name</td> <td>Opportunity</td> </tr> <tr> <td>Data type</td> <td>Master Detail relationship Object : Opportunity</td> </tr> </table>		Field Name	Opportunity	Data type	Master Detail relationship Object : Opportunity
Field Name	Opportunity						
Data type	Master Detail relationship Object : Opportunity						

Milestone 6 : Page Layouts :

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

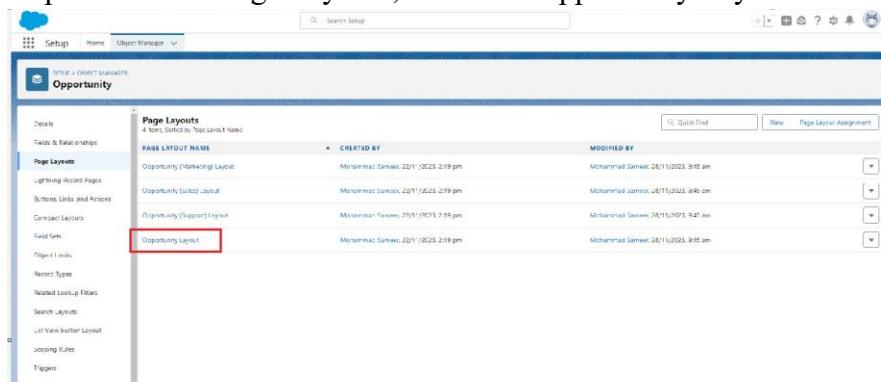
Use Case:

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

Activity 1: Edit the Page layout for Opportunity Object

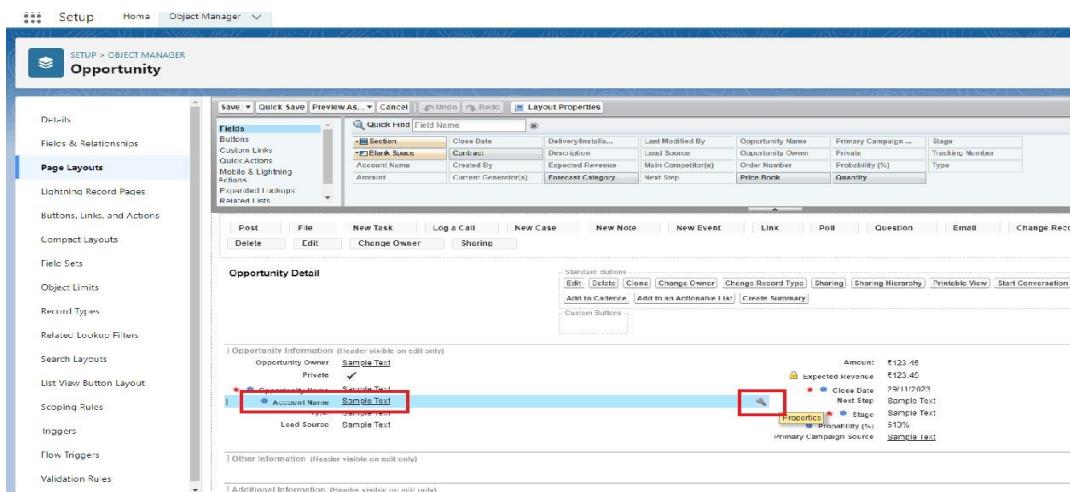
Step 1: Go to Setup → Click on Object Manager → On the search bar, select Opportunity Layout. You can notice Page Layouts on the left panel

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

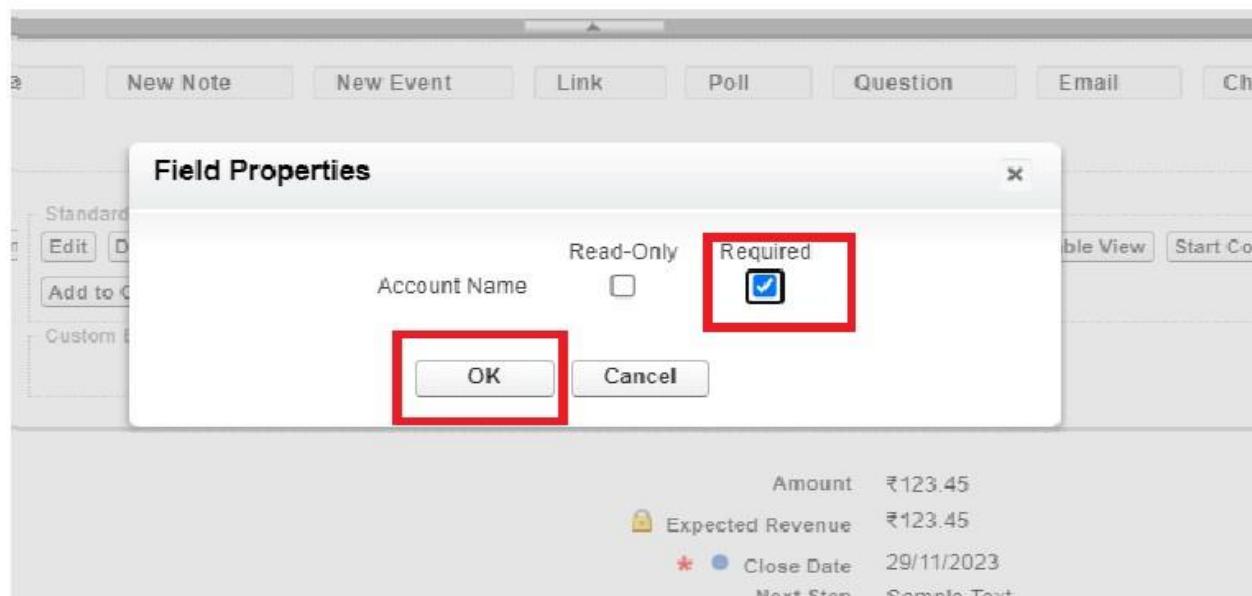


The screenshot shows the 'Opportunity' page layout configuration. On the left, there's a sidebar with various setup options like 'Fields & Relationships', 'Buttons, Links, and Actions', and 'Page Layouts'. The main area displays a table of page layouts. One row, 'Opportunity Layout', is highlighted with a red box. The table columns include 'Page Layout Name', 'Created By', and 'Modified By'.

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.



This screenshot shows the 'Layout Properties' for the 'Opportunity' object. The left sidebar lists 'Page Layouts' under 'Fields & Relationships'. The main area shows the 'Opportunity Detail' section with various fields. The 'Account Name' field is highlighted with a red box. Below it, a properties panel shows details like 'Amount: ₹123.45' and 'Close Date: 29/11/2023'.



A modal dialog titled 'Field Properties' is open, showing settings for the 'Account Name' field. The 'Required' checkbox is checked and highlighted with a red box. At the bottom, there are 'OK' and 'Cancel' buttons. The background shows the 'Opportunity Detail' section with other fields like 'Amount' and 'Close Date'.

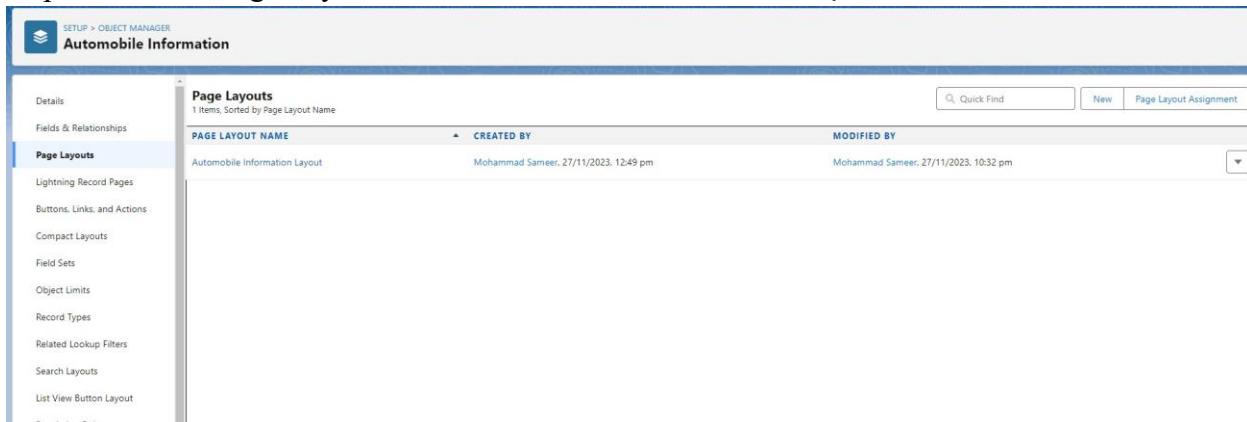
Step 4: check the Required box for Account name and click on Ok.

Step 5: Click on Save.

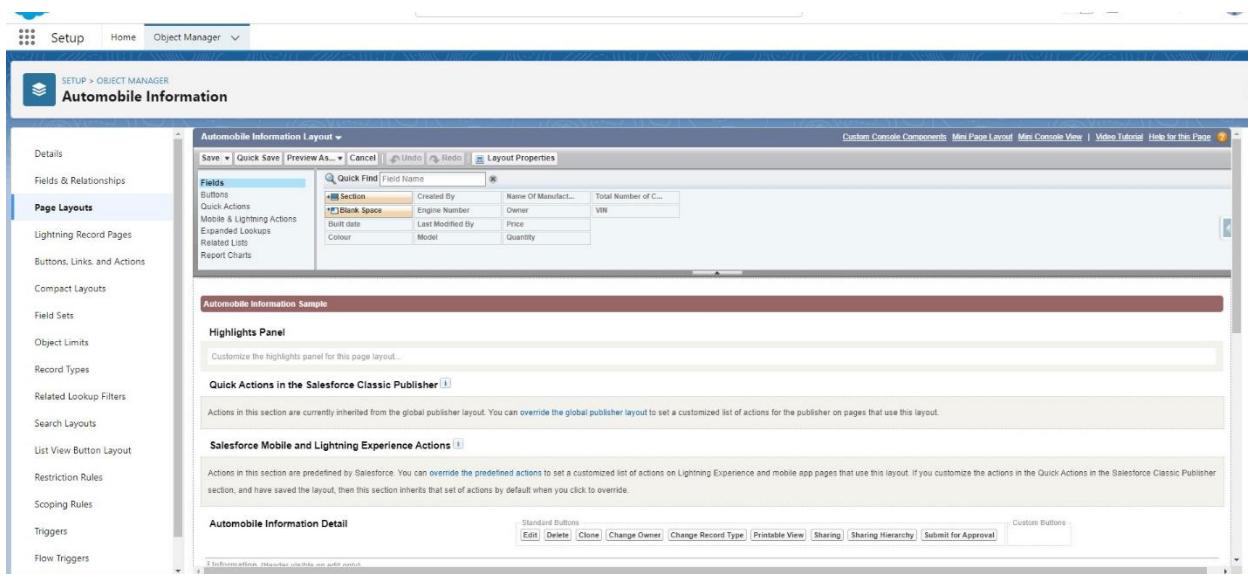
Activity 2: Edit the Page layout for Automobiles Information

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’.



PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Automobile Information Layout	Mohammad Sameer, 27/11/2023, 12:49 pm	Mohammad Sameer, 27/11/2023, 10:32 pm



Fields

Buttons	Created By	Name Of Manufact...	Total Number of C...
Quick Actions	Engine Number	Owner	VIN
Mobile & Lightning Actions	Built Date	Last Modified By	Price
Expanded Lookups	Colour	Model	Quantity
Related Lists			
Report Charts			

Automobile Information Sample

Highlights Panel

Customize the highlights panel for this page layout...

Quick Actions in the Salesforce Classic Publisher

Actions in this section are currently inherited from the global publisher layout. You can override the global publisher layout to set a customized list of actions for the publisher on pages that use this layout.

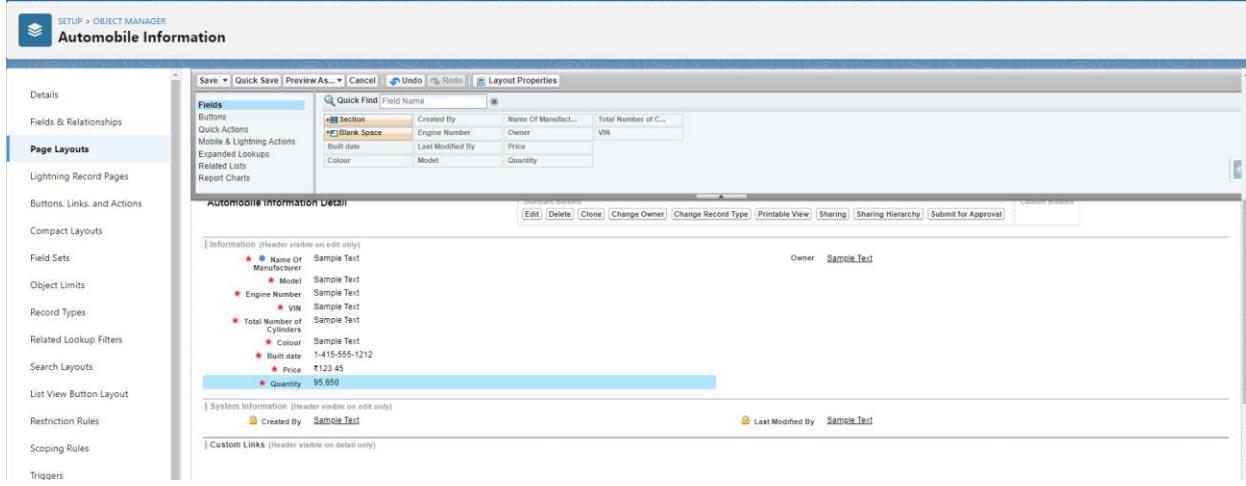
Salesforce Mobile and Lightning Experience Actions

Actions in this section are predefined by Salesforce. You can override the predefined actions to set a customized list of actions on Lightning Experience and mobile app pages that use this layout. If you customize the actions in the Salesforce Classic Publisher section, and have saved the layout, then this section inherits that set of actions by default when you click to override.

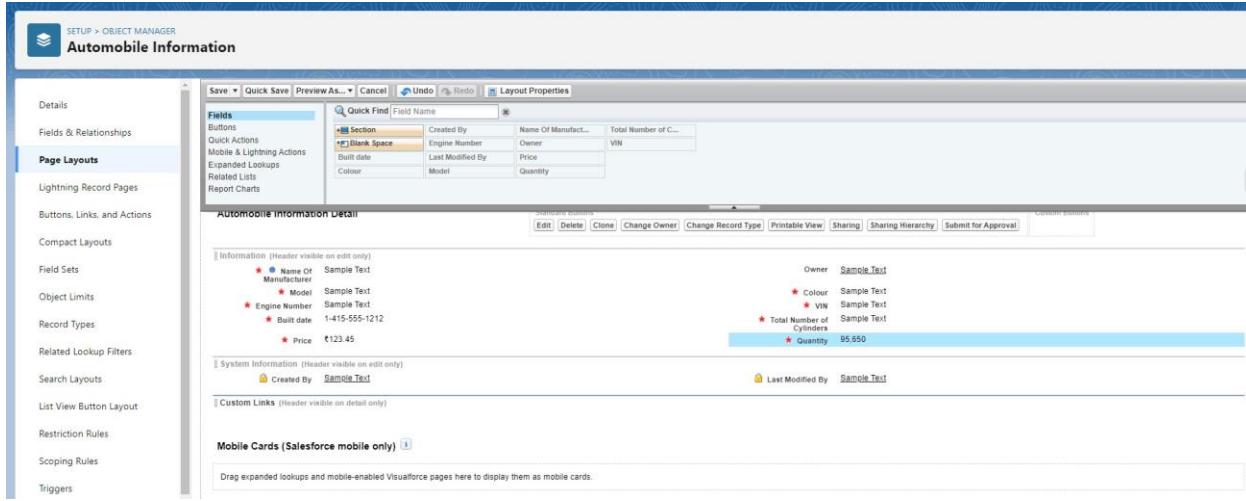
Automobile Information Detail

Standard Buttons: Edit, Delete, Clone, Change Owner, Change Record Type, Printable View, Sharing, Sharing Hierarchy, Submit for Approval, Custom Buttons

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.



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



Step 5 : Click on Save.

Milestone 7: Apex Trigger

Apex can be invoked by using triggers. Apex triggers enable you to perform custom actions before or after changes to Salesforce records, such as insertions, updates, or deletions. A trigger is Apex code that executes before or after the following types of operations:

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

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

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

There are primarily two types of Apex Triggers:

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

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

Activity- 1:

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

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

```
public class OpportunityHandlerClass {

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

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

        set<Id> AutoInformationIds = new set<Id>();
        for(Opportunity_Automobile__c OppAuto: lstOpportunityAutomobile.values()){
            if(OppAuto.Automobile__c != null){
                AutoInformationIds.add(OppAuto.Automobile__c);
            }
        }
        List<Automobile_Information__c> lstAutomobileInfomation = new List<Automobile_Information__c>();
        Map<Id,Automobile_Information__c> MapAutomobileInformation = New Map<Id,Automobile_Information__c>([SELECT Quantity__c, Price__c, Name, Id
        FROM Automobile_Information__c
        WHERE Id IN: AutoInformationIds]);
        For(Opportunity_Automobile__c AutoOpp : lstOpportunityAutomobile.Values()){
            decimal num = 0;
            if(AutoOpp.Automobile__c == MapAutomobileInformation.get(AutoOpp.Automobile__c).Id && OldMapOpportunity.get(AutoOpp.Opportunity__c).stagename != 'Closed Won'){

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

Code:

```
public class OpportunityHandlerClass {

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

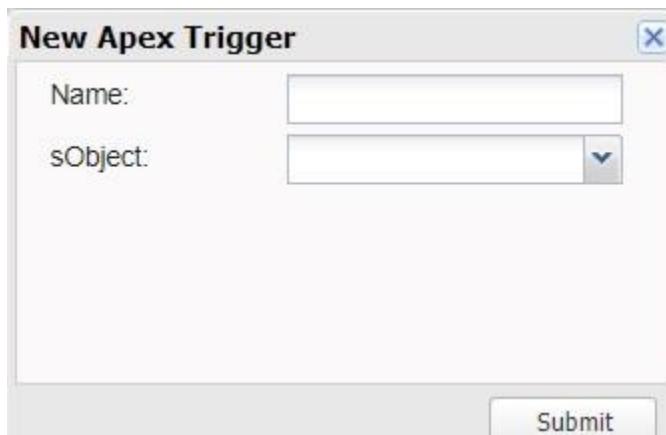
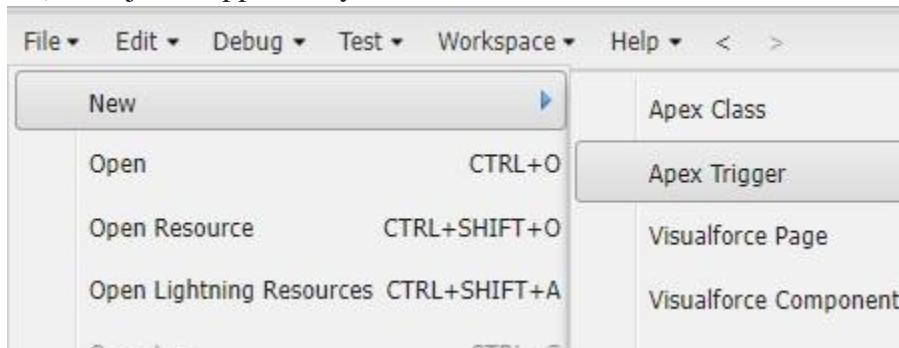
}
}

```

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 :

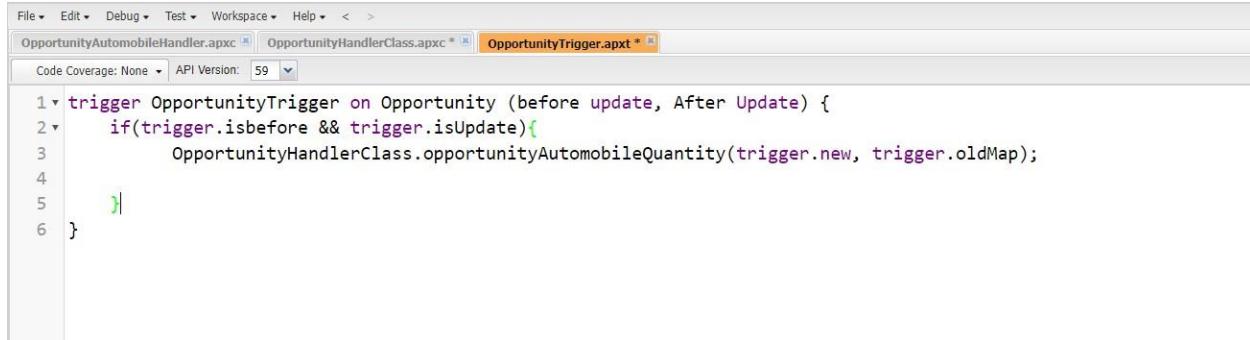
The syntax for creating trigger is :

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

}

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.



```

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >
OpportunityAutomobileHandler.apxc [ ] OpportunityHandlerClass.apxc * [ ] OpportunityTrigger.apxt *
Code Coverage: None ▾ API Version: 59 ▾
1 trigger OpportunityTrigger on Opportunity (before update, After Update) {
2     if(trigger.isbefore && trigger.isUpdate){
3         OpportunityHandlerClass.opportunityAutomobileQuantity(trigger.new, trigger.oldMap);
4     }
5 }
6

```

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

Activity- 2:

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

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

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



```

1 public class OpportunityAutomobileHandler {
2     public static void quantityErrorOnAutomobileInformation(List<Opportunity_Automobile__c> lstOpportunityAutomobile){
3         set<Id> AutomobileIds = new Set<Id>();
4         for(Opportunity_Automobile__c OppAutomobile : lstOpportunityAutomobile){
5             if(OppAutomobile.Automobile__c != null){
6                 AutomobileIds.add(OppAutomobile.Automobile__c);
7             }
8         }
9         Map<Id, Automobile_Information__c> lstAutomobileInformation = new map<Id, Automobile_Information__c>([SELECT Id, CreatedById, Quantity__c, Price__c
10                                         FROM Automobile_Information__c WHERE Id IN: AutomobileIds]);
11         for(Opportunity_Automobile__c OppAutomobile : lstOpportunityAutomobile){
12             if(OppAutomobile.Automobile__c == lstAutomobileInformation.get(OppAutomobile.Automobile__c).Id && lstAutomobileInformation.get(OppAutomobile.Automobile__c).Quantity__c < OppAutomobile.Quantity__c){
13                 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);
    }
}

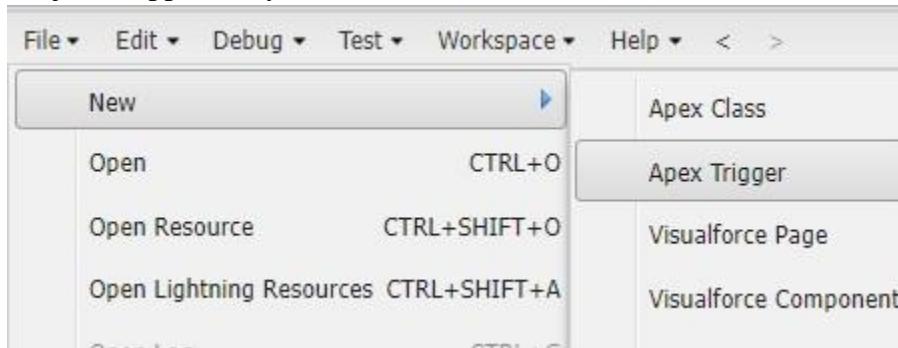
Map<Id,Automobile_Information__c> lstAutomobileInformation = new
map<Id,Automobile_Information__c>([SELECT Id, CreatedById, Quantity__c, Price__c FROM
Automobile_Information__c WHERE Id IN: AutomobileIds]);

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 :

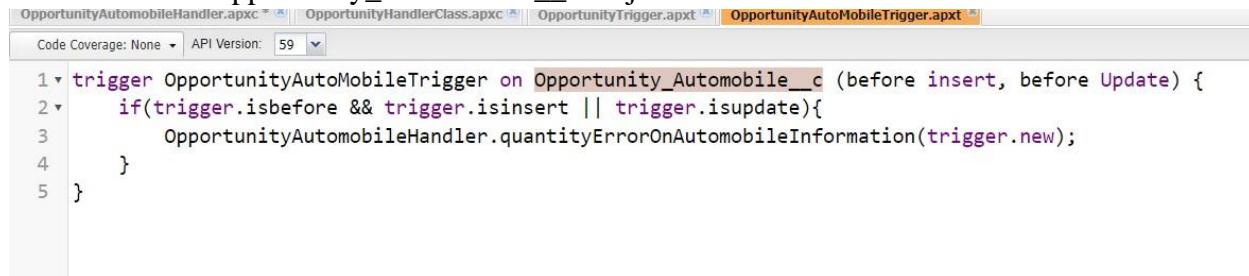
1. While still in the trailhead account, navigate to the gear icon in the top right corner.
2. Click on developer console and you will be navigated to a new console window.
3. Click on the File menu in the toolbar, and click on new → Trigger.
4. Enter the trigger name and the object to be triggered.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





Trigger :

Handler for the Opportunity_Automobile__c Object



```

trigger OpportunityAutoMobileTrigger on Opportunity_Automobile__c (before insert, before update) {
    if(trigger.isbefore && trigger.isinsert || trigger.isupdate){
        OpportunityAutomobileHandler.quantityErrorOnAutomobileInformation(trigger.new);
    }
}

```

Code: trigger OpportunityAutoMobileTrigger on Opportunity_Automobile__c (before insert, before update) {
 if(trigger.isbefore && trigger.isinsert || trigger.isupdate){
 OpportunityAutomobileHandler.quantityErrorOnAutomobileInformation(trigger.new);
 }
}
}

Activity- 3 :

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

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

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

```

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

```

```

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

```

Trigger Handler :

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

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

Open

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

Code Coverage: None | API Version: 59

```

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

Trigger:

```

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

Activity- 4 :

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

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

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

```

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

```

Trigger:

```

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

```

Trigger Handler :

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

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

Open					
Entity Type	Entities		Related		
	Name	Number of	Name	Extent	Direction
Entity Type			← Opportunity...	ApexClass	References
Classes	OpportunityTrigger		← Opportunity	SObject	References
Triggers	OpportunityAutoMobile...				
Pages					
Page Components					
Objects					
Static Resources					
Packages					

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:

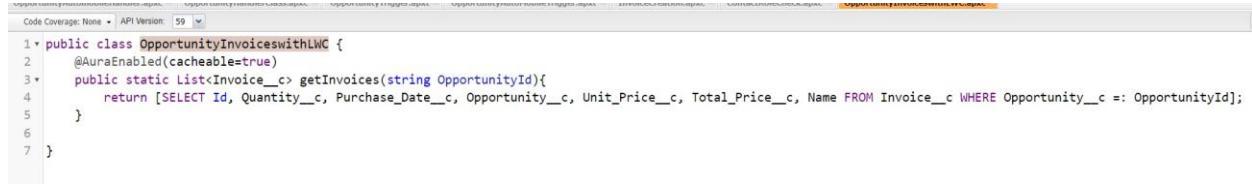
```

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);
    }
}
```

Milestone 8: LWC Component:

Activity- 1 : Create Apex Class to Get Invoices

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



```
Code Coverage: None • API Version: 59
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];
    }
}
```

Activity- 1: 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
  .
```

Activity- 2 : 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.

Activity- 3: Install the Salesforce Extension Pack

In the VS Code,

1. go to extensions (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.



EXTENSIONS: MARKETPLACE

salesforce extension pack 2

Salesforce Extension ... ⚡ 947K ★ 3
Extensions for developing on the Sa...
↳ Salesforce **Install**

Extension Pack for J... ⚡ 16.4M ★ 4
Popular extensions for Java develop...
↳ Microsoft **Install**

C/C++ Extension P... ⚡ 12.1M ★ 4.5
Popular extensions for C++ develop...
↳ Microsoft **Install**

Python Extension Pack ⚡ 4.2M ★ 4
Popular Visual Studio Code extensio...
Don Jayamanne **Install**

PHP Extension Pack ⚡ 3.6M ★ 4.5
Everything you need for PHP develo...
Xdebug **Install**

Salesforce Extension ... ⚡ 155K ★ 5
Extensions for developing on the Sa...
↳ Salesforce **Install**

Live Share Extension... ⚡ 2.5M ★ 4.5

Salesforce Extension Pack v56.5.1

Salesforce | ⚡ 9,47,031 | ★★★★☆(43)

Extensions for developing on the Salesforce Platform

Install

Details Changelog

Extension Pack (9)

- Salesforce CLI Integration**
Provides integration with the Salesforce CLI
↳ Salesforce **Install**
- Apex**
Provides code-editing features for the Apex ...
↳ Salesforce **Install**
- Aura Components**
Provides code-editing features for Aura Com

The extension pack is installed successfully

Extension: Salesforce Extension Pack X

Salesforce Extension Pack v56.5.1

↳ Salesforce | ⚡ 9,47,031 | ★★★★☆(43)

Extensions for developing on the Salesforce Platform

Disable **Uninstall** **⚙️**

This extension is enabled globally.

Install the Salesforce Extension Pack

Activity- 4 : Create a project in VS Code

1. Press CTRL + SHIFT + P, type sf: create
2. select SFDX: Create Project with Manifest

Get Started - Visual Studio Code

>sfdx: create

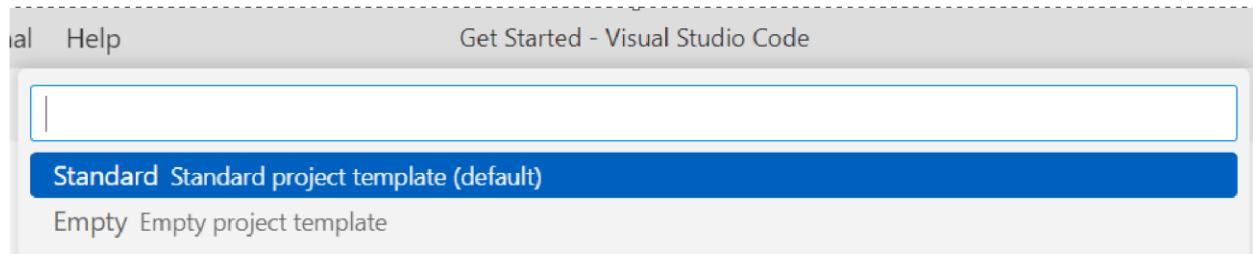
SFDX: Create Project with Manifest

SFDX: Create and Set Up Project for ISV Debugging

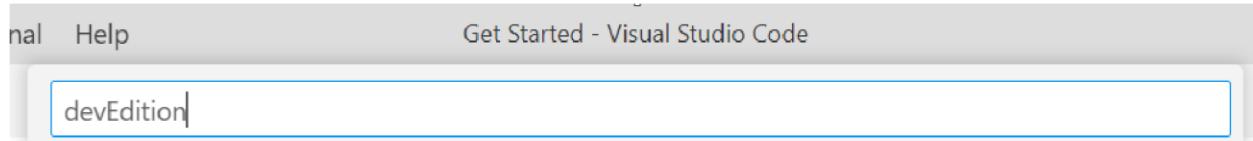
recently used ⚙️

other commands

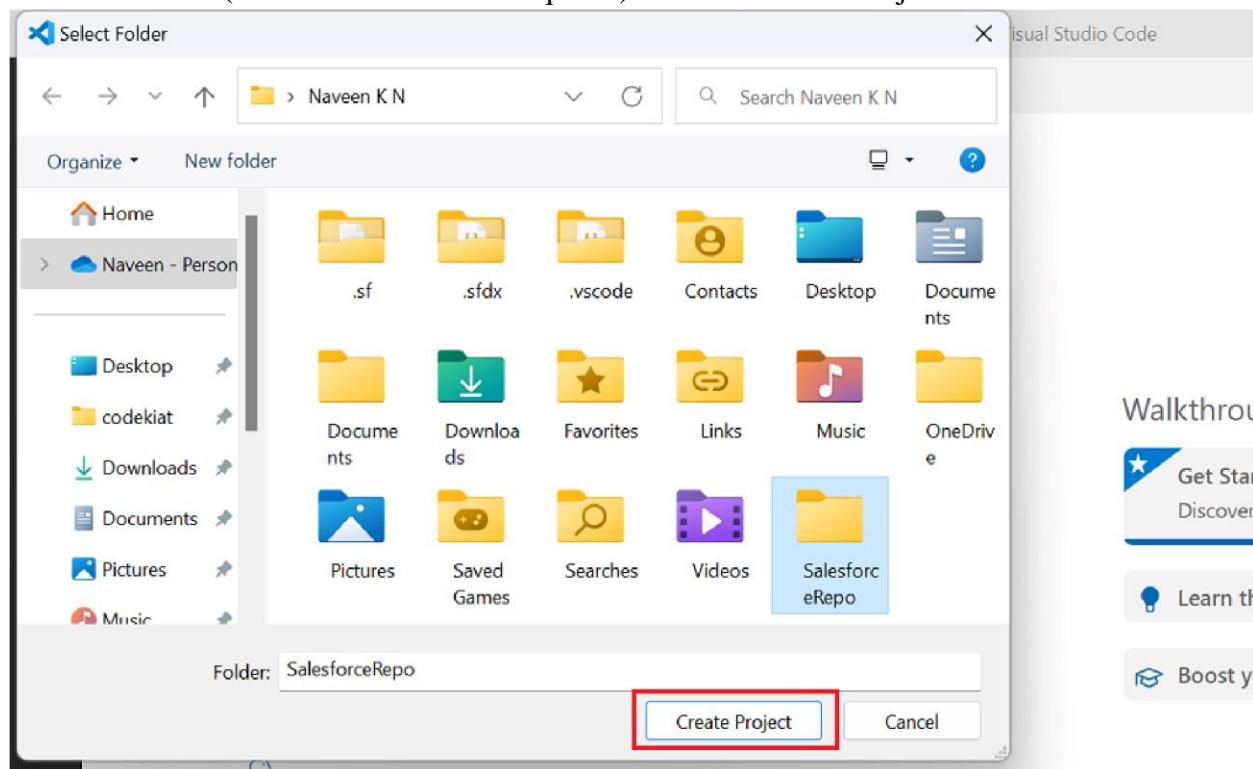
3. Select the Standard project template



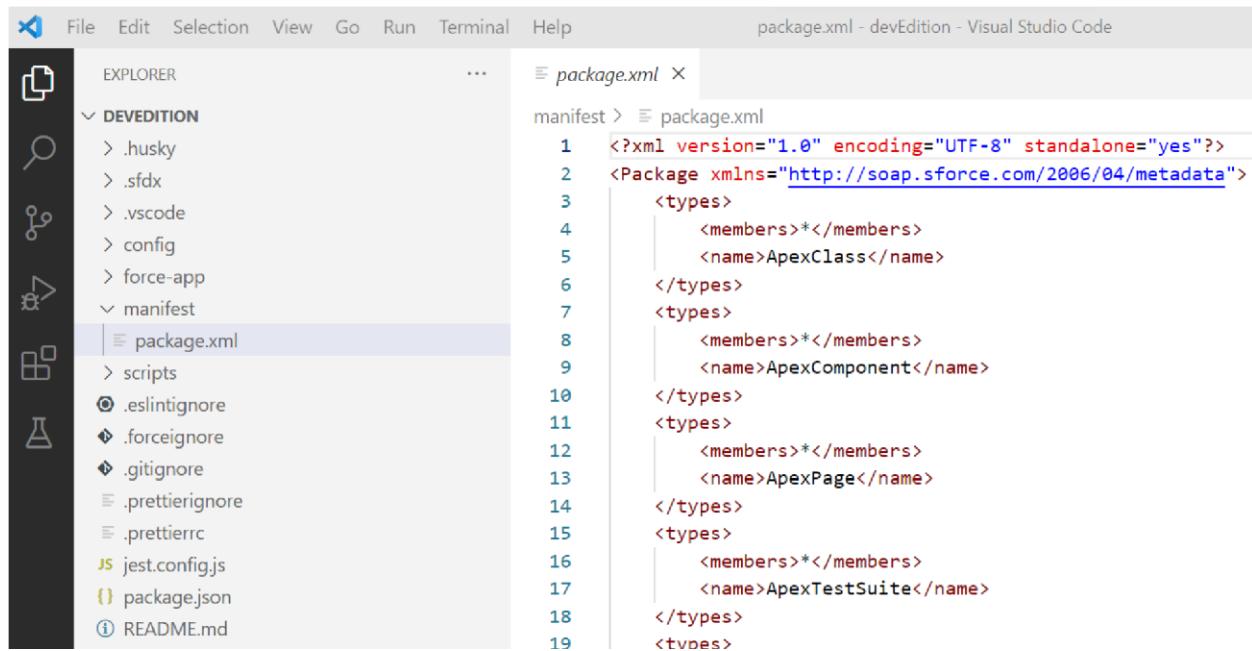
4. Type a project name and Click Enter.



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



6. The new project is created with package.xml



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays a project structure under 'DEVEDITION'. The 'package.xml' file is selected in the list. The main area shows the XML code for the package:

```

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

```

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

```

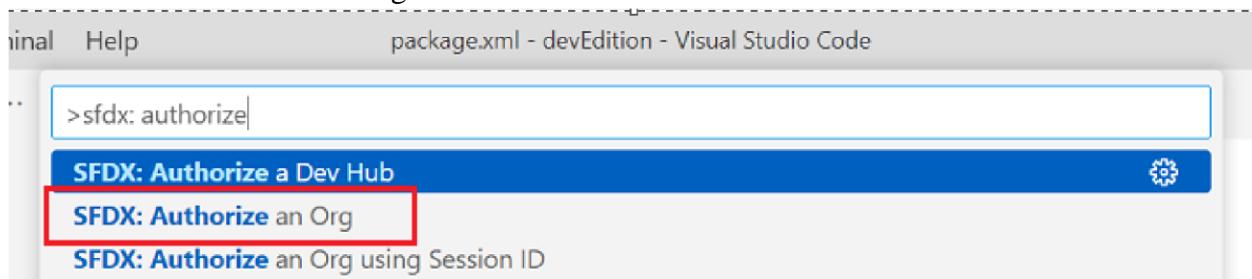
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Package xmlns="http://soap.sforce.com/2006/04/metadata">
  <types>
    <members*></members>
    <name>LightningComponentBundle</name>
  </types>
  <version>55.0</version>
</Package>

```

Activity- 5 : Authorize an org

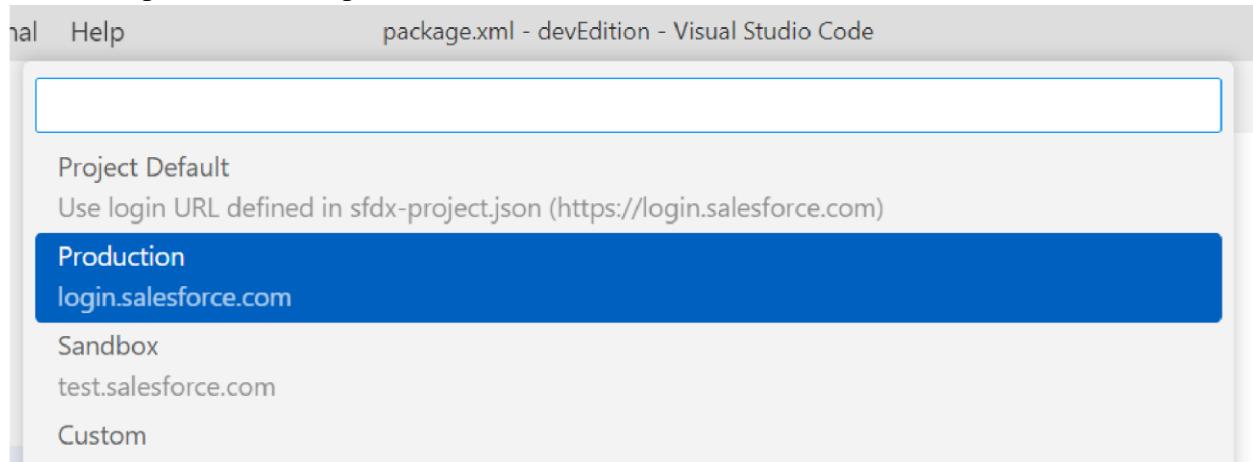
Establish a connection between the local project and the Salesforce instance to retrieve and deploy the components.

1. Press CTRL + SHIFT + P, type sfdx: authorize.
2. select SFDX: Authorize an Org from the list



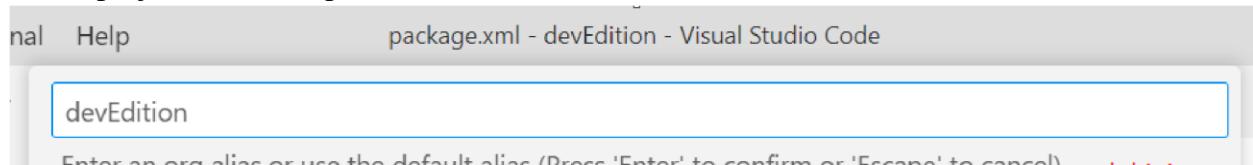
3. Choose your Salesforce instance.

For developer edition and production instances select Production.



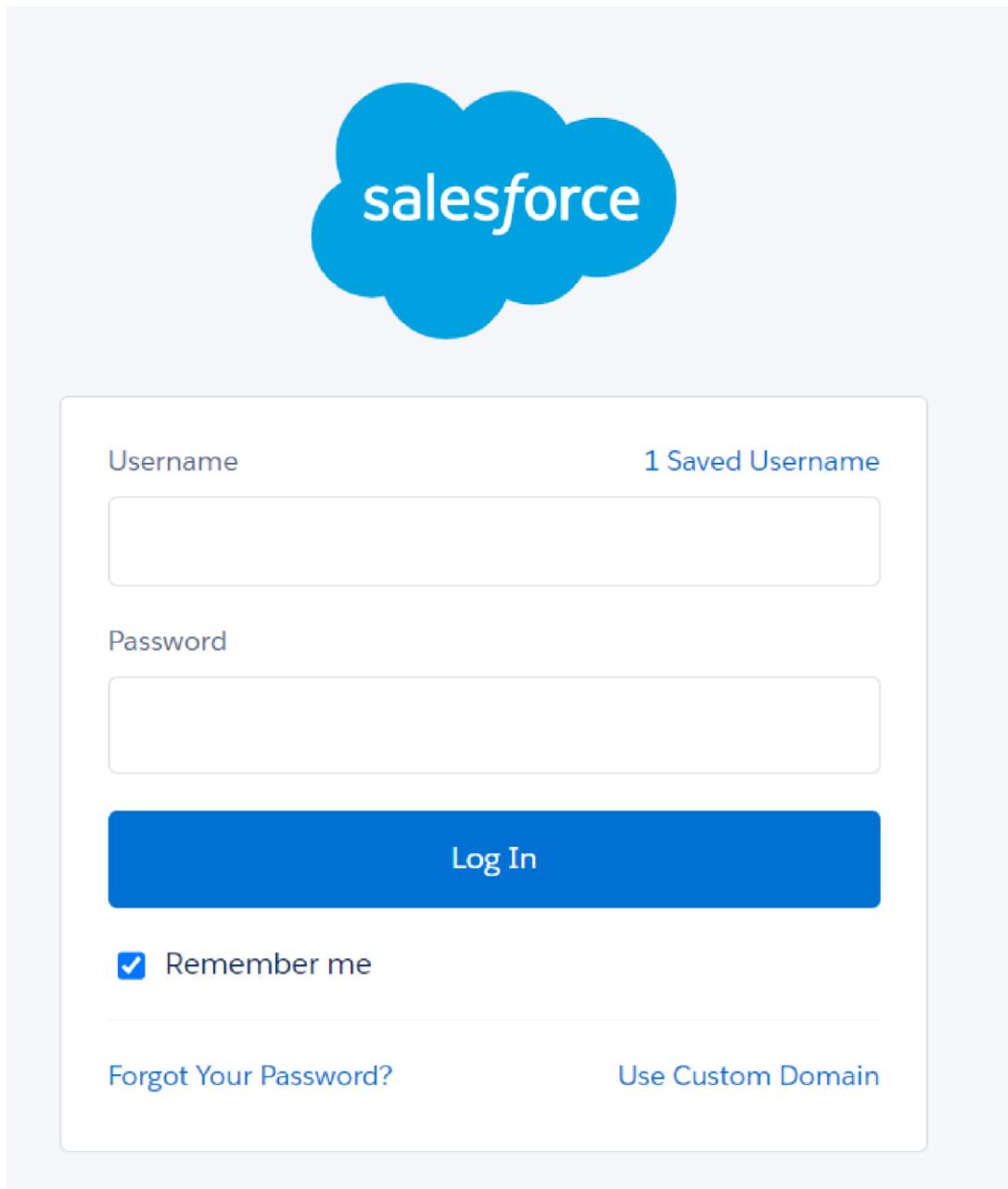
4. For this demonstration, I used the developer edition, hence it is Production.

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

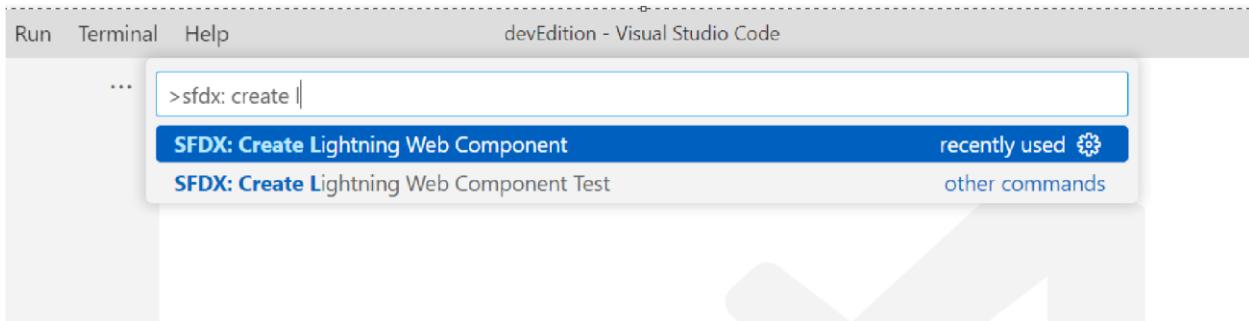


8. It will be successfully authorized.

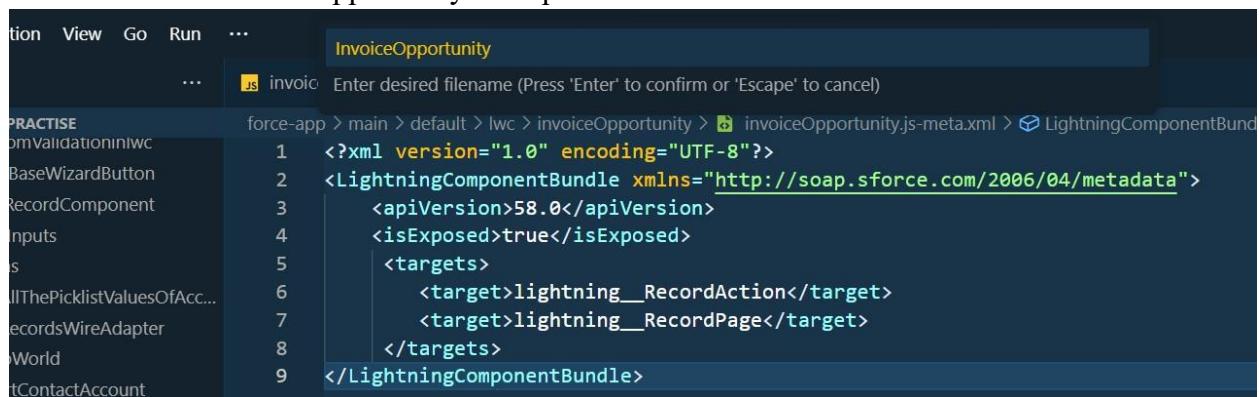
Activity- 6 : Create Lightning Web Component

XML File :

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



2. Give the name “InvoiceOpportunity” and press Enter.

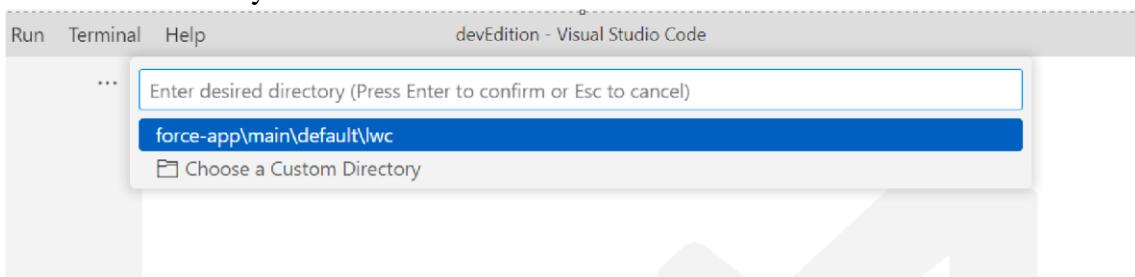


```

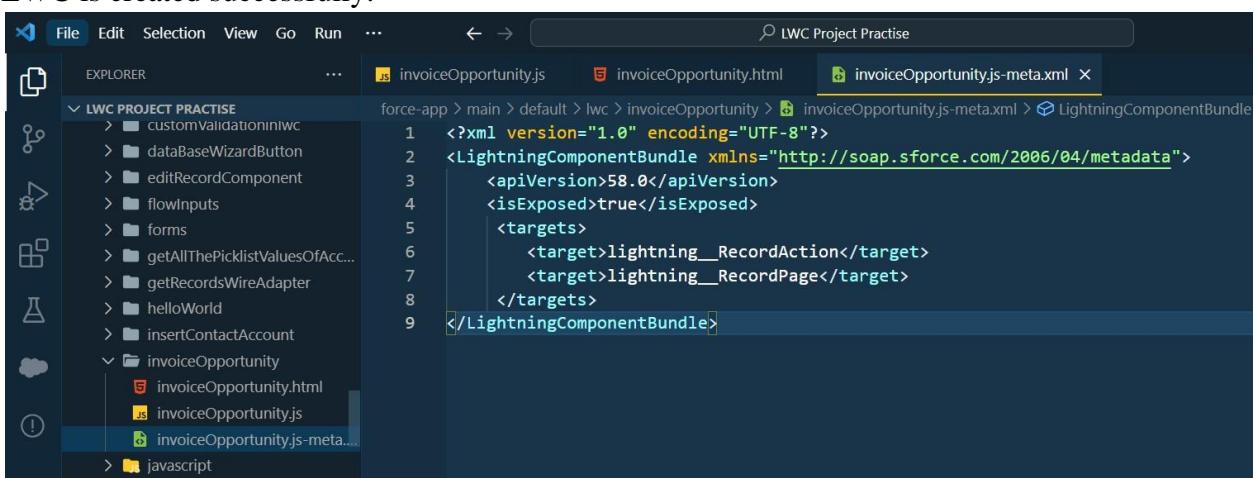
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>58.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordAction</target>
        <target>lightning__RecordPage</target>
    </targets>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

```

3. Choose the directory.



4. LWC is created successfully.



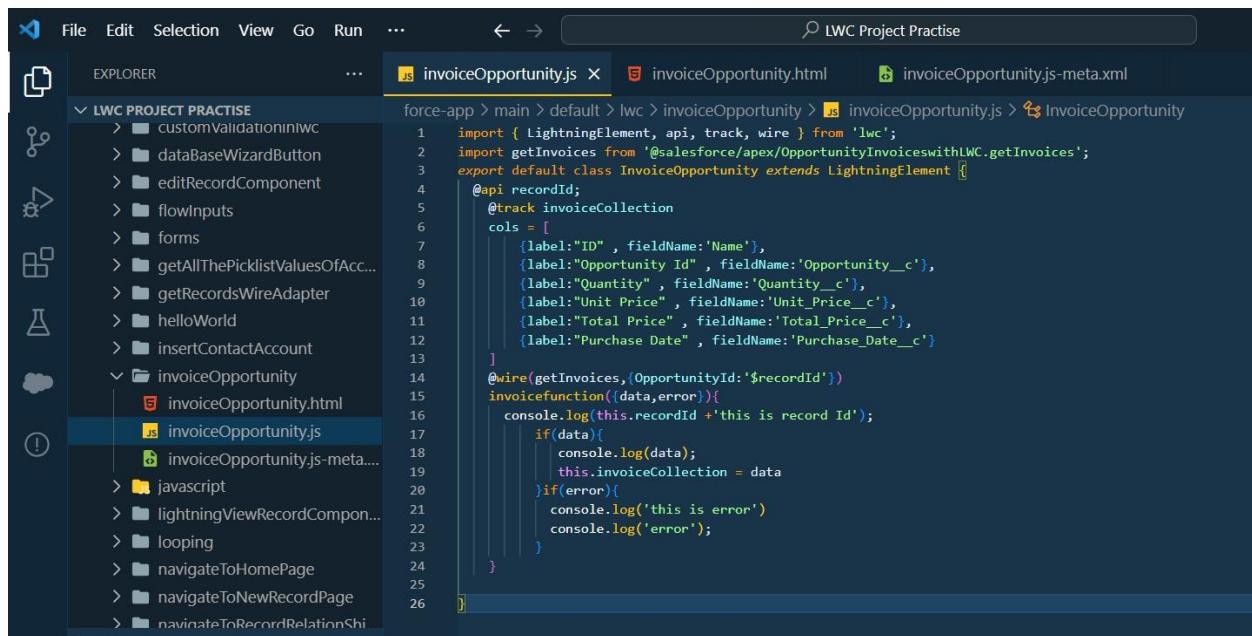
5. Copy and paste the below-mentioned code in the InvoiceOpportunity.js-meta.xml and update the apiVersion tag with the latest API version.

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 Salesforce LWC Project Practise interface. On the left, the Explorer sidebar displays a project structure under 'LWC PROJECT PRACTISE' with files like 'customValidationinlwc', 'dataBaseWizardButton', 'editRecordComponent', 'flowInputs', 'forms', 'getAllThePicklistValuesOfAcc...', 'getRecordsWireAdapter', 'helloWorld', 'insertContactAccount', and 'invoiceOpportunity'. Within 'invoiceOpportunity', there are 'invoiceOpportunity.html', 'invoiceOpportunity.js', and 'invoiceOpportunity.js-meta....'. The main editor area is titled 'invoiceOpportunity.js' and contains the following code:

```
1 import { LightningElement, api, track, wire } from 'lwc';
2 import getInvoices from '@salesforce/apex/OpportunityInvoiceswithLWC.getInvoices';
3 export default class InvoiceOpportunity extends LightningElement {
4     @api recordId;
5     @track invoiceCollection
6     cols = [
7         {label:'ID' , fieldName:'Name'},
8         {label:'Opportunity Id' , fieldName:'Opportunity__c'},
9         {label:'Quantity' , fieldName:'Quantity__c'},
10        {label:'Unit Price' , fieldName:'Unit_Price__c'},
11        {label:'Total Price' , fieldName:'Total_Price__c'},
12        {label:'Purchase Date' , fieldName:'Purchase_Date__c'}
13    ]
14    @wire(getInvoices,{OpportunityId:$recordId})
15    invoicefunction({data,error}){
16        console.log(this.recordId +'this is record Id');
17        if(data){
18            console.log(data);
19            this.invoiceCollection = data
20        }if(error){
21            console.log('this is error')
22            console.log(error);
23        }
24    }
25}
26}
```

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;
```

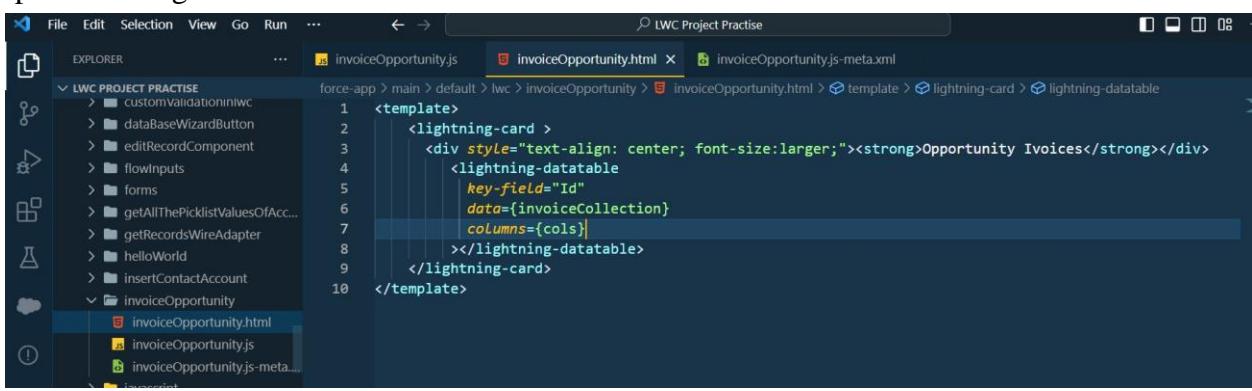
```

@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>

```

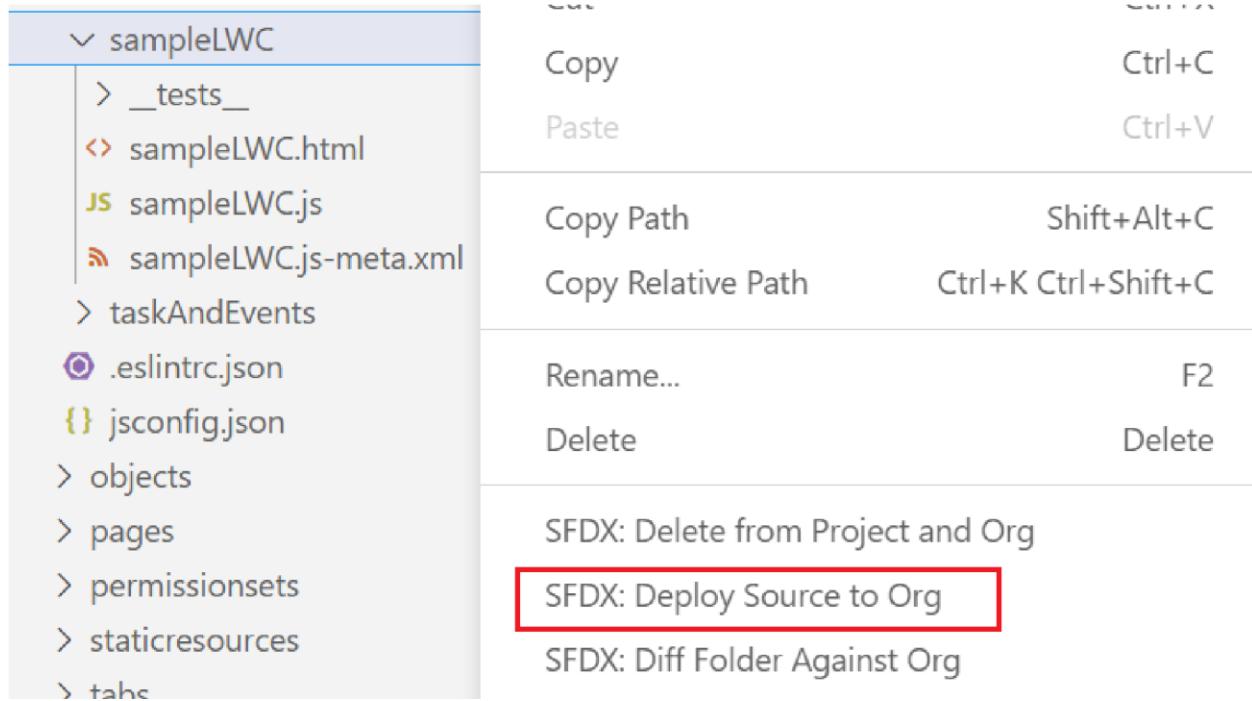
HTML File Code:

```
<template>
```

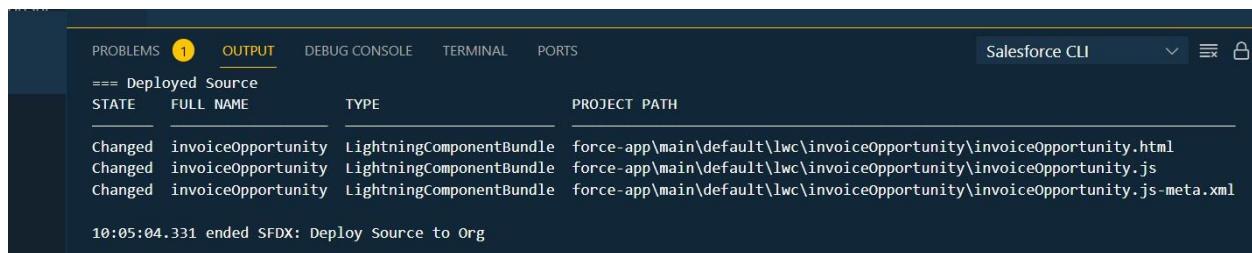
```
<lightning-card>
    <div style="text-align: center; font-size:larger;"><strong>Opportunity
    Ivvoices</strong></div>
    <lightning-datable key-
        field="Id"
        data={invoiceCollection}
        columns={cols}
    ></lightning-datable>
</lightning-card>
</template>
```

Deploy Component:

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

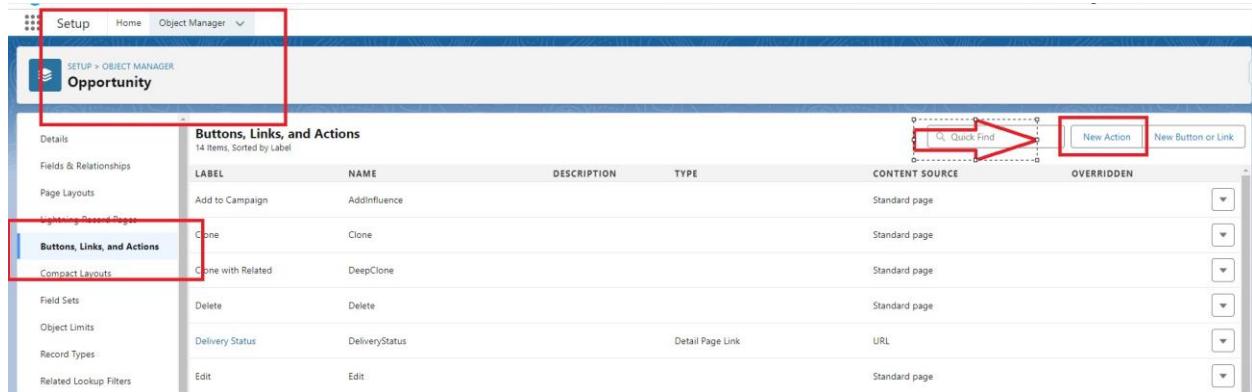


2. Once the deployment is complete, you will see the below-highlighted message in the output tab



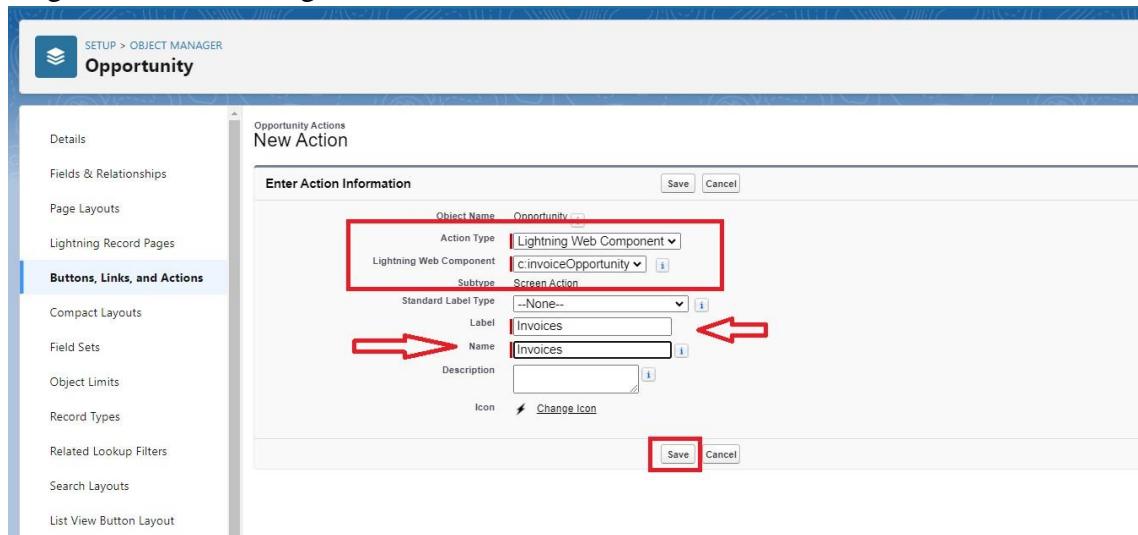
Activity- 7 : Create Button to Add on Opportunity

1. To add the newly created component to the view, Go to Salesforce Setup
2. Click on Object Manager 3. Search Opportunity and Click on it .
4. click on Button Links and Action.
5. click on the New Action.



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		

6. Select Action type as Lightning Web Component
7. Select the InvoiceOpportunity component
 - a. Label :- Invoices
 - b. Name :- Invoices
8. As given on below image



Opportunity Actions
New Action

Enter Action Information

Action Type: Lightning Web Component

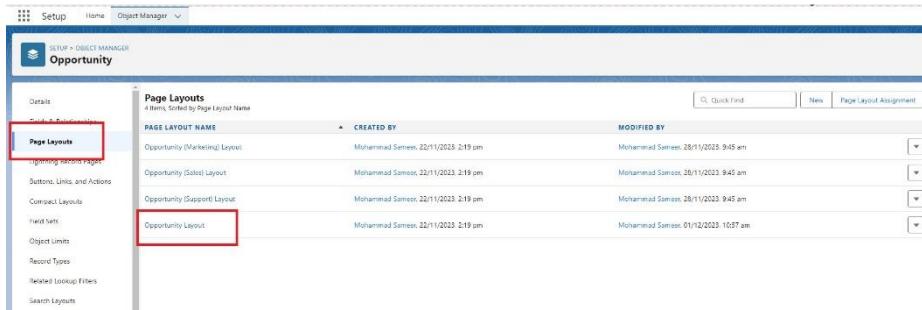
Label: Invoices

Name: Invoices

9. Click on Save and your action Button is Ready.

Activity- 8 : Add InvoiceOpportunity into Opportunity Record Page

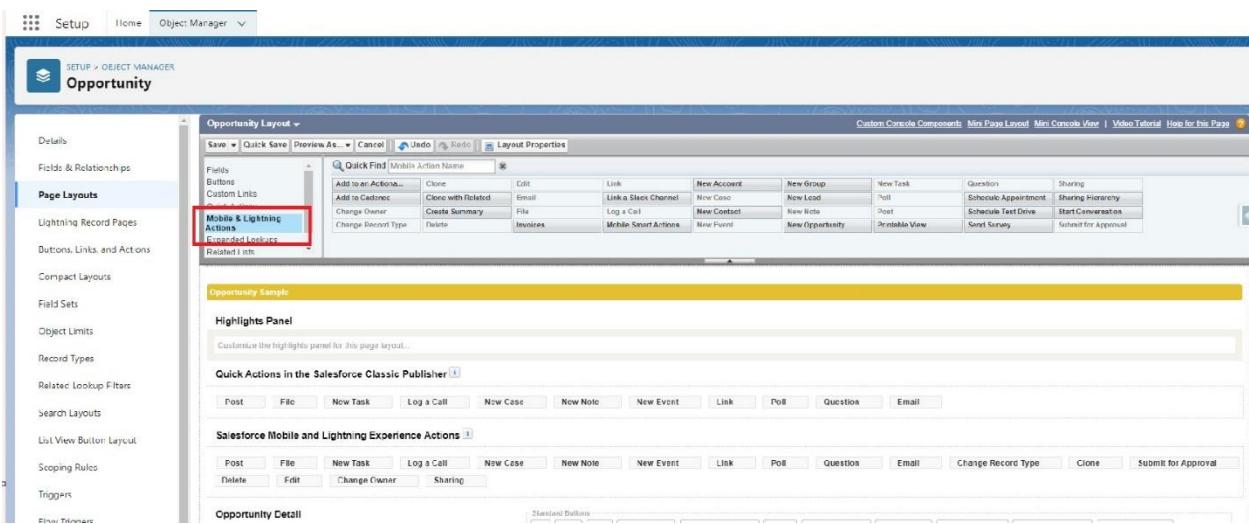
1. On Opportunity Object Manager Click on Page layout.
2. Click on OpportunityLayout.



The screenshot shows the 'Object Manager' interface for the 'Opportunity' object. In the left sidebar, under 'Page Layouts', the 'Opportunity Layout' is selected and highlighted with a red box. The main area displays a table of page layouts, with the 'Opportunity Layout' entry also highlighted with a red box. The table includes columns for 'Page Layout Name', 'Created By', and 'Modified By'.

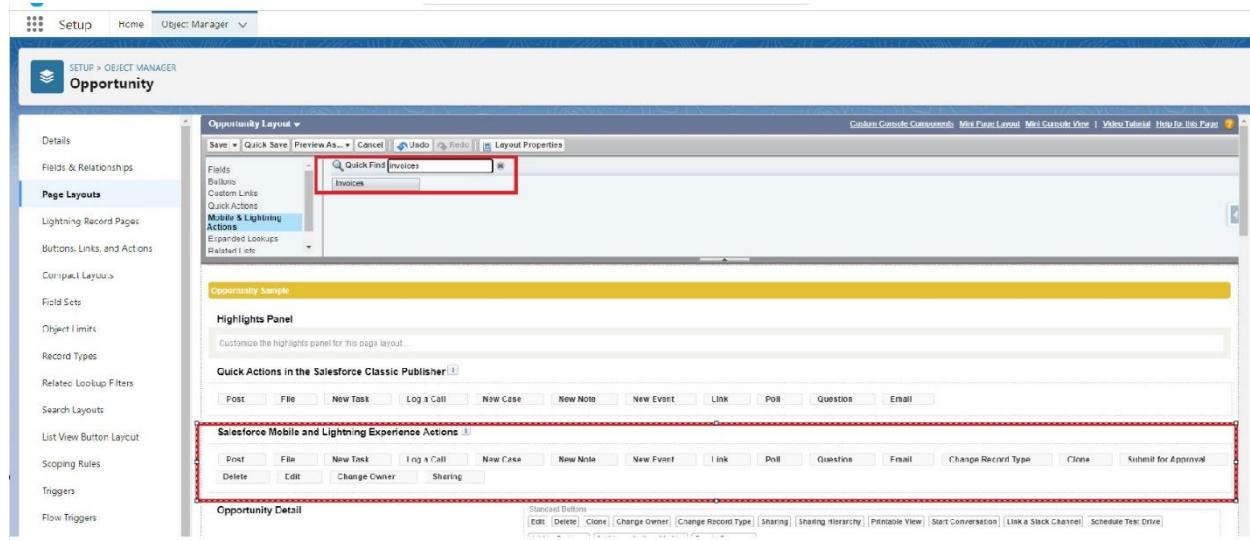
PAGE LAYOUT NAME	CREATED BY	MODIFIED BY
Opportunity Marketing Layout	Mohammad Samer, 22/11/2023, 2:19 pm	Mohammad Samer, 28/11/2023, 9:45 am
Opportunity (Global) Layout	Mohammad Samer, 22/11/2023, 2:19 pm	Mohammad Samer, 20/11/2023, 9:45 am
Opportunity Support Layout	Mohammad Samer, 22/11/2023, 2:19 pm	Mohammad Samer, 20/11/2023, 9:45 am
Opportunity Layout	Mohammad Samer, 22/11/2023, 2:19 pm	Mohammad Samer, 01/12/2023, 10:57 am

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



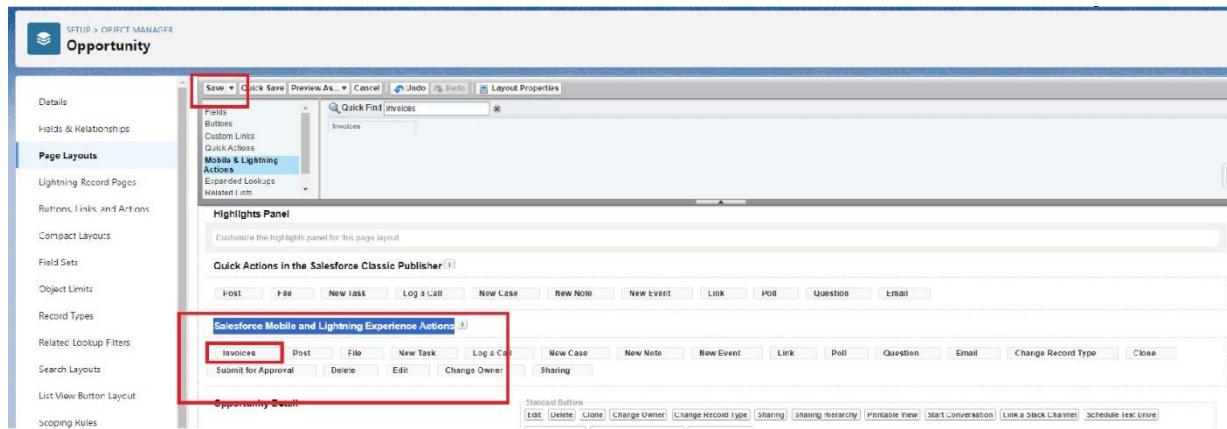
The screenshot shows the 'Opportunity Layout' editor. In the left sidebar, under 'Page Layouts', the 'Mobile & Lightning Actions' button is highlighted with a red box. The main area contains several sections: 'Fields & Relationships', 'Opportunity Layout', 'Highlights Panel', 'Quick Actions in the Salesforce Classic Publisher', 'Salesforce Mobile and Lightning Experience Actions', and 'Opportunity Detail'.

4. Search for invoice on Quick Find.



The screenshot shows the Salesforce setup interface for managing object layouts. The left sidebar lists various configuration options like Details, Fields & Relationships, Page Layouts, and Page Layouts (under Lightning Record Pages). The main area is titled 'Opportunity Layout' and contains sections for 'Fields & Relationships' (with a red box highlighting the 'Mobile & Lightning Actions' dropdown), 'Opportunity Sample' (containing a 'Highlights Panel' and 'Quick Actions' section), and 'Opportunity Detail' (containing a 'Standard Buttons' bar). The top navigation bar includes 'Save', 'Quick Save', 'Preview As...', 'Cancel', 'Undo', 'Redo', and 'Layout Properties'.

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



This screenshot shows the same setup interface after performing the drag-and-drop action. The 'Mobile & Lightning Actions' section has been moved into the 'Salesforce Mobile and Lightning Experience Actions' section. A red box highlights the 'Save' button at the top left of the layout editor. The rest of the interface remains the same, showing the 'Opportunity Sample' and 'Opportunity Detail' sections.

6. Click on Save.

Milestone 9 : Apex Schedulers :

The Apex Scheduler lets you delay execution so that you can run Apex classes at a specified time. This is ideal for daily or weekly maintenance tasks using Batch Apex. To take advantage of the scheduler, write an Apex class that implements the Schedulable interface, and then schedule it for execution on a specific schedule.

Schedulable Apex Syntax :

To invoke Apex classes to run at specific times, first implement the Schedulable interface for the class. Then, schedule an instance of the class to run at a specific time using the System.schedule() method.

After you implement a class with the Schedulable interface, use the System.schedule() method to execute it. The System.schedule() method uses the user's timezone for the basis of all schedules, but runs in system mode—all classes are executed, whether or not the user has permission to execute the class.

SYNTAX :

```
public class SomeClass implements Schedulable {  
    public void execute(SchedulableContext ctx) {  
        // awesome code here  
    }  
}
```

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 :



The screenshot shows a Salesforce code editor with the file `DeleteClosedLostOpportunities.apxc` selected. The code implements a `Schedulable` interface:

```

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 }

```

```

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.

apex classes

Custom Code

Apex Classes

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

Setup Home Object Manager ▾

Apex Class

SETUP Apex Classes

Apex Classes

Help for this Page

Percent of Apex Used: 0.14%
You are currently using 8,427 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 |

Compile all classes |

View: All Create New View

Action	Name	Namespace Prefix	Api Version	Status	Size Without Comments	Last Modified By	Has Trace Flags
Edit Del Security	ContactRoleCheck		59.0	Active	672	Mohammad Sameer	29/11/2023, 10:13 am
Edit Del Security	DeleteClosedLostOpportunities		59.0	Active	356	Mohammad Sameer	01/12/2023, 11:57 am
Edit Del Security	InvoiceCreation		59.0	Active	1,243	Mohammad Sameer	29/11/2023, 10:05 am
Edit Del Security	OpportunityAutomobileHandler		59.0	Active	1,065	Mohammad Sameer	29/11/2023, 10:12 am
Edit Del Security	OpportunityHandleClass		59.0	Active	4,041	Mohammad Sameer	29/11/2023, 9:49 am
Edit Del Security	OpportunityInvoiceWithLWC		59.0	Active	319	Mohammad Sameer	29/11/2023, 3:04 pm

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Other | All

Dynamic Apex Classes

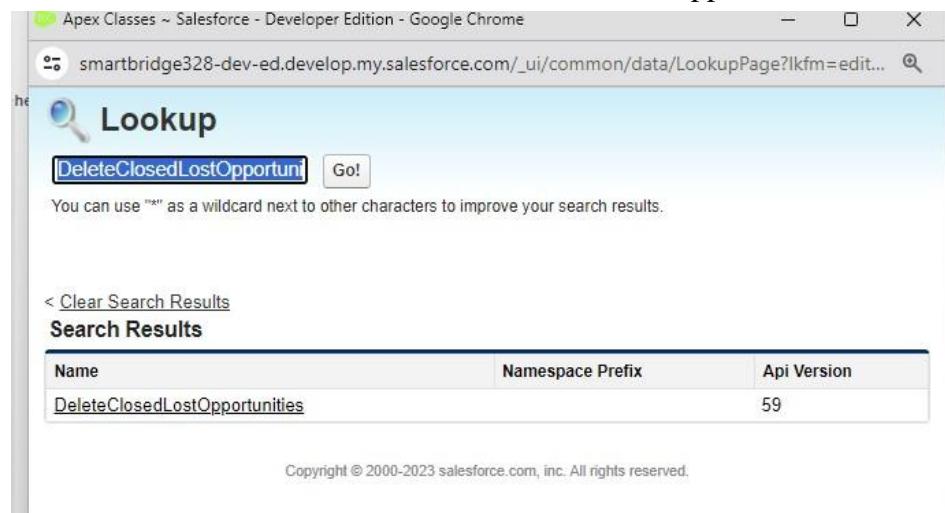
- Click on Schedule Apex and enter the Job name.
 - Job Name : DeleteOpportunitySchedule

Schedule Apex

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

		Save	Cancel
Job Name	<input type="text" value="DeleteOpportunitySchedule"/>		
Apex Class	<input type="text" value="DeleteClosedLostOpportunities"/> 		
Schedule Apex Execution <div style="display: flex; justify-content: space-between;"> <div style="flex: 1;"> <p>Frequency</p> <input checked="" type="radio"/> Weekly <input type="radio"/> Monthly </div> <div style="flex: 1;"> <p>Recurs every week on</p> <p><input type="checkbox"/> Sunday</p> <p><input checked="" type="checkbox"/> Monday</p> <p><input type="checkbox"/> Tuesday</p> <p><input type="checkbox"/> Wednesday</p> <p><input type="checkbox"/> Thursday</p> <p><input type="checkbox"/> Friday</p> <p><input type="checkbox"/> Saturday</p> </div> </div> <div style="margin-top: 10px;"> <p>Start <input type="text" value="01/12/2023"/> <input]<="" p="" type="text" value="01/12/2023"/> <p>End <input type="text" value="01/01/2024"/> <input]<="" p="" type="text" value="01/12/2023"/> <p>Preferred Start Time <input type="text" value="10:00 am"/></p> </p></p></div> <p style="text-align: center; margin-top: 10px;">Exact start time will depend on job queue activity.</p>			

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



The screenshot shows a browser window titled "Apex Classes ~ Salesforce - Developer Edition - Google Chrome". The URL is "smartbridge328-dev-ed.develop.my.salesforce.com/_ui/common/data/LookupPage?lkfm=edit...". The main content is a "Lookup" page for the class "DeleteClosedLostOpportunities". The search bar contains "DeleteClosedLostOpportuni". Below the search bar, there is a message: "You can use '*' as a wildcard next to other characters to improve your search results." There is a link "[Clear Search Results](#)". The "Search Results" section shows a table with one row:

Name	Namespace Prefix	Api Version
DeleteClosedLostOpportunities		59

At the bottom of the page, there is a copyright notice: "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: DeleteClosedLostOpportuni

Schedule Apex Execution

Frequency: Weekly

Recur every week on:

- Sunday
- Monday
- Tuesday
- Wednesday
- Thursday
- Friday
- Saturday

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.

3) Click on Save. Now enter Apex in the search box and select Apex 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, 10:00 am	Scheduled Apex
Del	Analytics Data Loader Job for Org : 005j00000DvWY	User_Integration	23/11/2023, 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.

Milestone 10 : Reports:

Reports give you access to your Salesforce data. You can examine your Salesforce data in almost infinite combinations, display it in easy-to-understand formats, and share the resulting insights with others. Before building, reading, and sharing reports, review these reporting basics.

Types of Reports in Salesforce

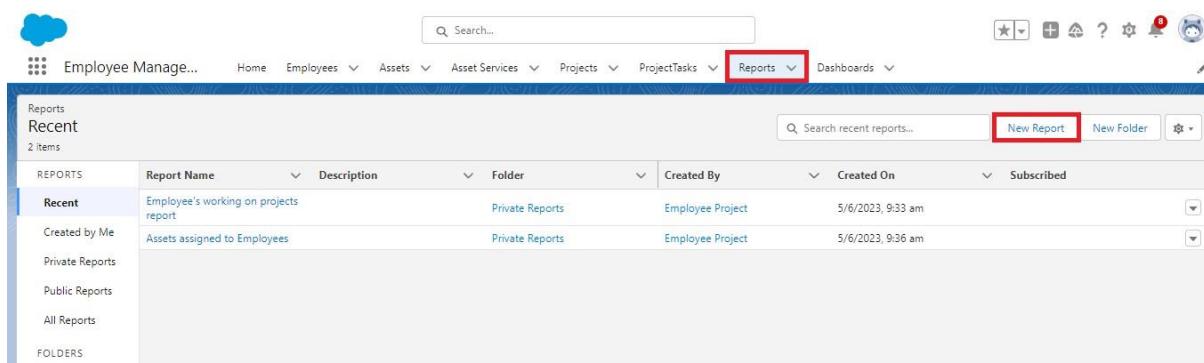
1. Tabular
2. Summary
3. Matrix
4. Joined Reports Use Case:

The CEO of an organization wants to have brief data on Opportunity Sales along with Invoices generated.

Let's create a Report.

Activity- 1 : Create Report on Opportunity

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



The screenshot shows the Salesforce Reports interface. At the top, there is a navigation bar with links for Home, Employees, Assets, Asset Services, Projects, ProjectTasks, Reports (which is currently selected and highlighted with a red box), and Dashboards. Below the navigation bar is a search bar labeled "Search...". On the left, there is a sidebar with sections for Reports (Recent, 2 items), Recent (Employee's working on projects report, Assets assigned to Employees), and Folders (Private Reports, Public Reports, All Reports). The main content area displays a table of recent reports. The table has columns for Report Name, Description, Folder, Created By, Created On, and Subscribed. The first two rows are visible: "Employee's working on projects report" (Folder: Private Reports, Created By: Employee Project, Created On: 5/6/2023, 9:33 am) and "Assets assigned to Employees" (Folder: Private Reports, Created By: Employee Project, Created On: 5/6/2023, 9:36 am). At the top right of the main content area, there are buttons for "New Report" (highlighted with a red box), "New Folder", and other report-related functions.

Report Name	Description	Folder	Created By	Created On	Subscribed
Employee's working on projects report		Private Reports	Employee Project	5/6/2023, 9:33 am	
Assets assigned to Employees		Private Reports	Employee Project	5/6/2023, 9:36 am	

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

Create Report

Category

- Recently Used
- All**
- Accounts & Contacts
- Opportunities
- Customer Support: Reports
- Leads
- Campaigns
- Activities
- Contracts and Orders
- Price Books, Products and Assets
- Administrative Reports
- File and Content Reports
- Individuals

Select a Report Type

Report Type Name	Category
Opportunities	Standard
Opportunities with Products	Standard
Opportunities with Contact Roles	Standard
Opportunities with Partners	Standard
Opportunities with Competitors	Standard
Opportunities with Contact Roles and Products	Standard
Opportunities with Opportunity Automobiles	Standard
Opportunities with Opportunity Automobiles and Automobile	Standard
Opportunities with Invoice	Standard
Campaigns with Opportunities	Standard
Campaigns with Influenced Opportunities	Standard
Activities with Opportunities	Standard
Products with Opportunities	Standard

Details

 **Opportunities**
Standard Report Type

Start Report

Details 

Fields (118)

Created By You
No Reports Yet

Created By Others
No Reports Yet

Objects Used in Report Type

-  Account
-  Profile
-  Campaign
-  User

4. Customize your report

- Add fields from left pane as shown below

Opportunity Closed Won Report

Filters

Show Me All Opportunities

Close Date All Time

Opportunity Status Any

Probability All

Opportunities

Previews a limited number of records. Run the report to see everything.

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

Employee Management

Reports

New Employees Report

Groups

GROUP ROWS

Add group...

Columns

Add column...

Employee: Employee Name X

Employee ID X

Reports to X

Login Time X

Logout Time X

Mode of Work X

Uncheck Profile X

Employees

Previews a limited number of records. Run the report to see everything.

Employee: Employee Name	Employee ID	Reports to	Login Time	Logout Time	Mode of Work	Uncheck Profile
Employee	A0C000000H9Y1	-	-	-	-	http://https://linkden.in
User for Internship	A0C000000H9Y1	-	8:00 am	9:00 pm	-	http://linkedIn

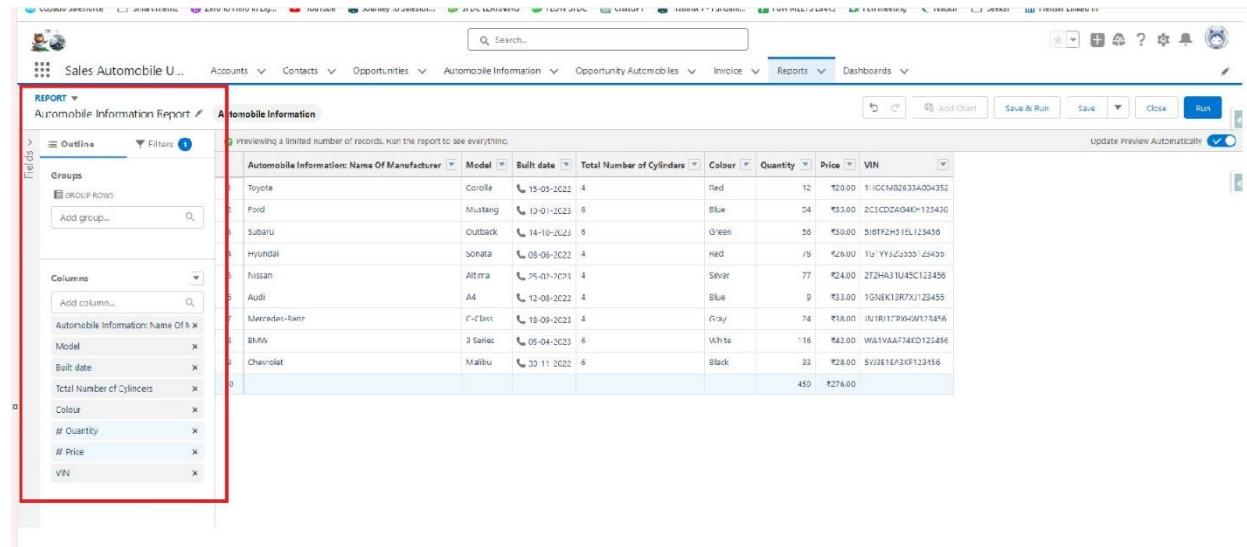
Add the Above Filter as well.

5. Save or run it.

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

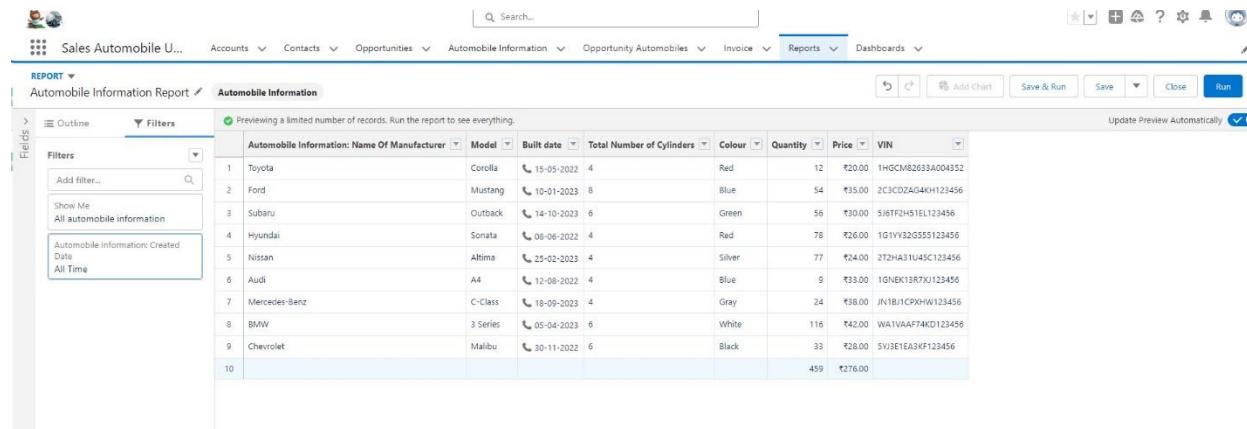
Activity- 2 : Create Report on Automobile Information.

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



Name Of Manufacturer	Model	Built date	Total Number of Cylinders	Colour	Quantity	Price	VIN
Toyota	Corolla	15-05-2022	4	Red	12	₹20,00	1HGCN82633A004352
Ford	Mustang	12-01-2023	8	Blue	54	₹35,00	2C3CDZAG04K123456
Subaru	Outback	14-10-2023	6	Green	56	₹30,00	5J6TF2HS1EL123456
Hyundai	Sonata	09-06-2022	4	Red	78	₹26,00	1U-YY523220-Z5420
Nissan	Altima	25-07-2021	4	Silver	77	₹24,00	2T2HA31U45C123456
Audi	A4	12-09-2022	4	Blue	9	₹33,00	1GNEK13R7XJ123456
Mercedes-Benz	C-Class	18-09-2023	4	Gray	74	₹38,00	JN1B1CP0HW123456
BMW	3 Series	05-04-2023	6	White	116	₹42,00	WA1VAA74KD123456
Chevrolet	Malibu	30-11-2022	6	Black	33	₹28,00	5V3E1EA3KF123456
					459	₹276,00	

Filters :-



Name Of Manufacturer	Model	Built date	Total Number of Cylinders	Colour	Quantity	Price	VIN
1 Toyota	Corolla	15-05-2022	4	Red	12	₹20,00	1HGCN82633A004352
2 Ford	Mustang	10-01-2023	8	Blue	54	₹35,00	2C3CDZAG04K123456
3 Subaru	Outback	14-10-2023	6	Green	56	₹30,00	5J6TF2HS1EL123456
4 Hyundai	Sonata	08-06-2022	4	Red	78	₹26,00	1GYY520555123456
5 Nissan	Altima	25-02-2023	4	Silver	77	₹24,00	2T2HA31U45C123456
6 Audi	A4	12-08-2022	4	Blue	9	₹33,00	1GNEK13R7XJ123456
7 Mercedes-Benz	C-Class	18-09-2023	4	Gray	24	₹38,00	JN1B1CP0HW123456
8 BMW	3 Series	05-04-2023	6	White	116	₹42,00	WA1VAA74KD123456
9 Chevrolet	Malibu	30-11-2022	6	Black	33	₹28,00	5V3E1EA3KF123456
10					459	₹276,00	

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

Milestone 11 :Dashboard:

Dashboards help you visually understand changing business conditions so you can make decisions based on the real-time data you've gathered with reports. Use dashboards to help users identify trends, sort out quantities, and measure the impact of their activities. Before building, reading, and sharing dashboards, review these dashboard basics.

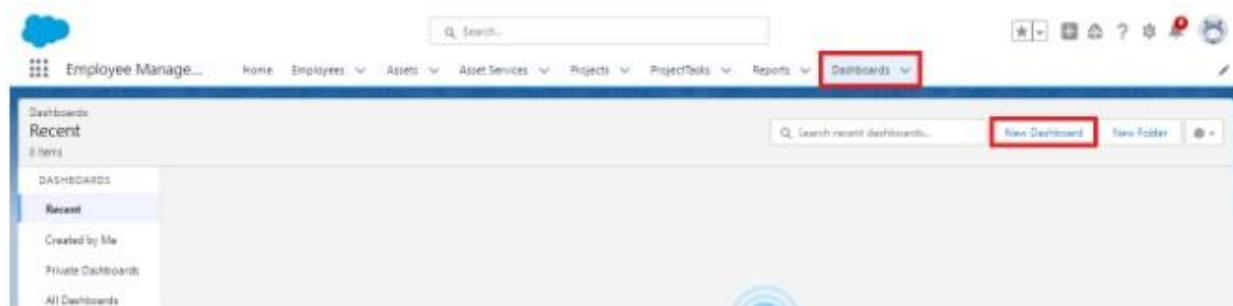
Use Case:

As an Admin for the organization you keep pushing yourself to reach out the business requirements to take the organization to peak heights and all your superiors are very much impressed with your efforts and work dedication. In addition with reports you make an ease for the CEO in viewing the reports with data visualization. So he doesn't have to search for the data he wants during the meetings.

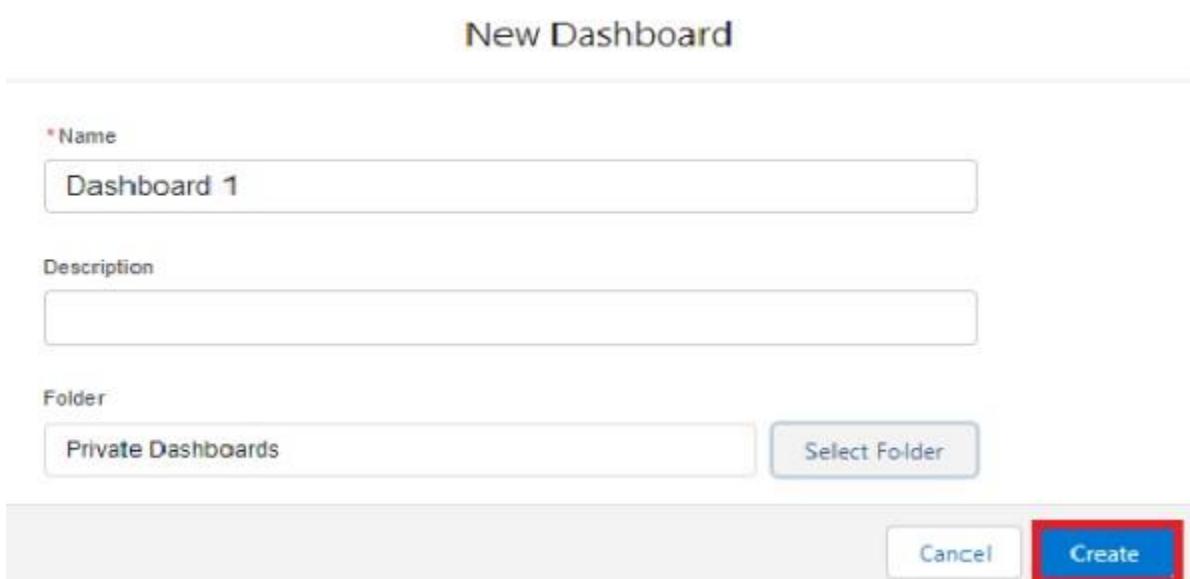
Activity 1:

Create Dashboard

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



2. Give a Name and click on Create.

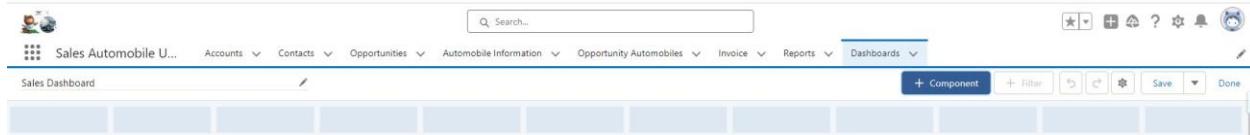


The 'New Dashboard' dialog box is displayed. It contains fields for 'Name' (with 'Dashboard 1' entered), 'Description' (an empty text area), 'Folder' (set to 'Private Dashboards'), and 'Select Folder' (a button). At the bottom are 'Cancel' and 'Create' buttons, with 'Create' being highlighted with a red box.

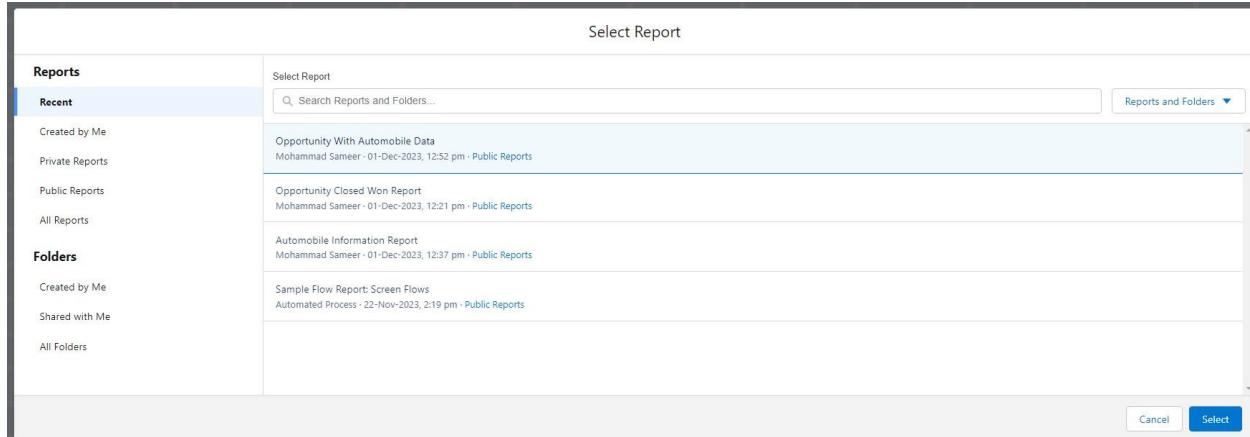
Name	Dashboard 1
Description	(empty)
Folder	Private Dashboards
Buttons: Cancel, Create	

Name : Automobile Sales

3. Select add component.

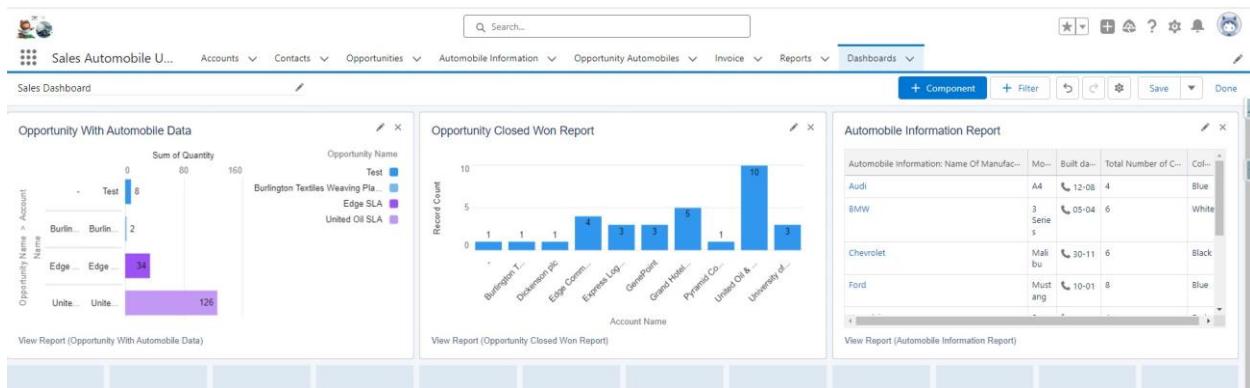


4. Select a Report and click on select.



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

The Created Dashboard will look like this.



Report Name	Record Count	Account Name
Opportunity With Automobile Data	160	Burlington Textiles Wearing Plan
Opportunity Closed Won Report	10	Dickenson Inc
Automobile Information Report	4	Edge Comm

5. Testing and Validation:

- **Unit Testing:**
 - ❖ Conducted tests for Apex classes and triggers to verify accuracy and reliability.
- **User Interface Testing:**
 - ❖ Ensured that all UI components function correctly across devices and roles.

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

- ❖ Lead-to-Opportunity Conversion: Streamlining the process of qualifying leads and moving them through the sales pipeline
- ❖ Automated Follow-Ups: Scheduling automated reminders for follow-ups to enhance customer engagement.
- ❖ Customer Insights: Providing actionable insights into customer preferences and buying patterns through analytics.
- ❖ Inventory Management: Monitoring and updating automobile inventory in real-time.

7. Conclusion:

The project successfully implemented a robust CRM system tailored to the needs of the automobile sales domain. It enhanced sales productivity, streamlined customer interactions, and provided insightful analytics for informed decision-making. By leveraging Salesforce, the project achieved its goal of transforming the automobile sales process into a more efficient and customer-centric system.

*****END*****