SALES AUTOMOBILE USING SALESFORCE

SALESFORCE PROJECT REPORT

PROJECT CREATED BY

MAGESH N	422422104055
ANEEQUA PARVEEN S	422422104056
RITHISH G	422422104057
JAYAPRIYA	422422104301

UNIVERSITY COLLEGE OF ENGINEERING TINDIVANAM

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ANNA UNIVERSITY CHENNAI 600025

November-2024

Table of contents

Project overview……3
Project objective……3
Key Features and concepts Utilized……..4
Detailed Step to Solution Design……..4
Testing and Validation……..27
Key Scenarios Addressed……....28
Conclusion……....29

Project Title: SALES AUTOMOBILE USING SALESFORCE

1. Project Overview

This project involves the development and implementation of a Customer Relationship Management (CRM) system tailored specifically for automobile sales. The goal is to streamline client management, enhance customer engagement, and optimize sales processes through Salesforce. By leveraging Salesforce's CRM platform, automobile dealerships can efficiently manage customer data, track interactions, and improve the customer experience from initial inquiry to post-purchase follow-up.

2. Project Objectives (Points Only)

- Enhance client relationship management.
- Optimize sales tracking and workflow.
- Improve data accessibility and integration.
- Provide real-time insights for decision-making.
- Increase efficiency in lead management and follow-up.

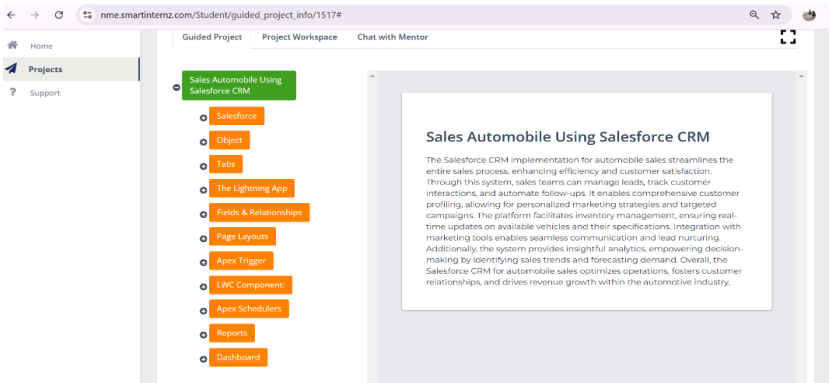

Fig:01

3. Salesforce Key Features and Concepts Utilized

In implementing this CRM application, several core Salesforce features and concepts were utilized:

- Salesforce CRM: Consolidates customer interactions, lead tracking, and account management.
- Salesforce Automation: Streamlines tasks like follow-ups and updates.
- Sales Cloud: Enabled for lead conversion, pipeline management, and sales forecasting specific to engineering project workflows.
- Custom Objects and Fields: Capture auto sales-specific data.
- Reports and Dashboards: Visual insights on sales performance and client engagement.
- Workflow Rules and Process Builder:Automate tasks, like notifying the sales team when a new lead enters.
- API Integrations: Sync data between Salesforce and other dealership applications.

4. Detailed Steps to Solution Design

Develop thorough documentation of the design, encompassing data models, user interface designs, and business logic. Ensure that all elements are accompanied by relevant screenshots.

Creating DeveloperAccount:

Creating a developer org in salesforce.

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

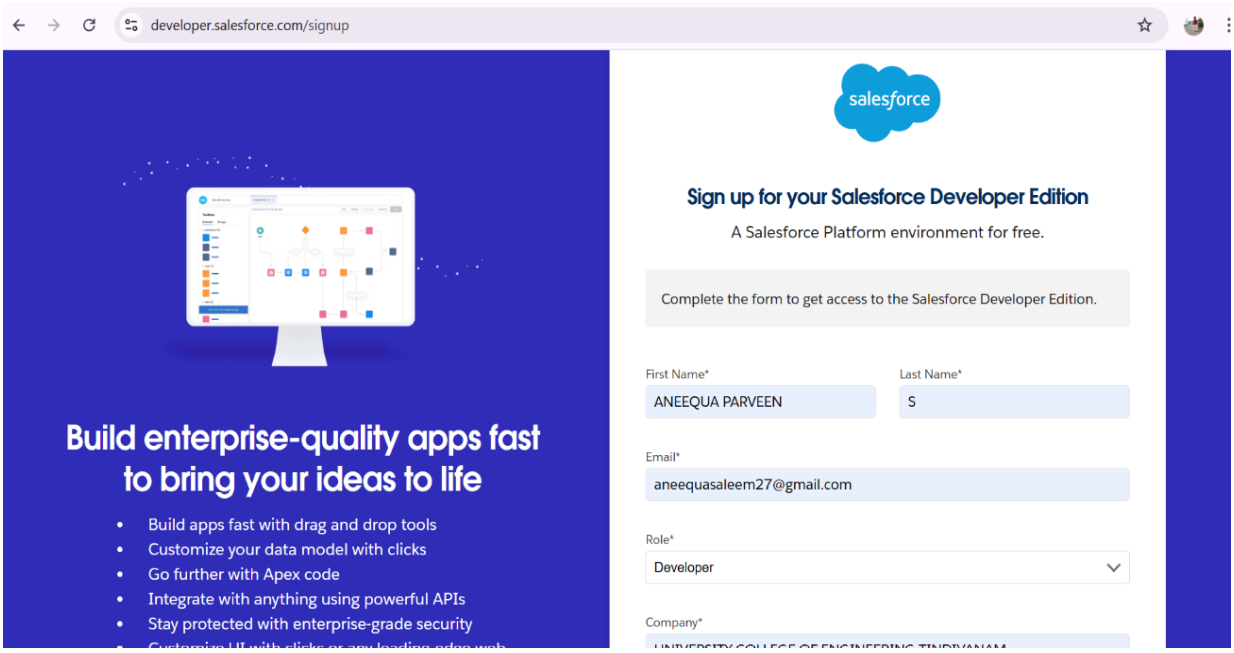

Fig:02

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

This need not be an actual email id, you can give anything in the format : username@organization.com

Click on sign me up after filling these.

➢ Account Activation:

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. Project Overview

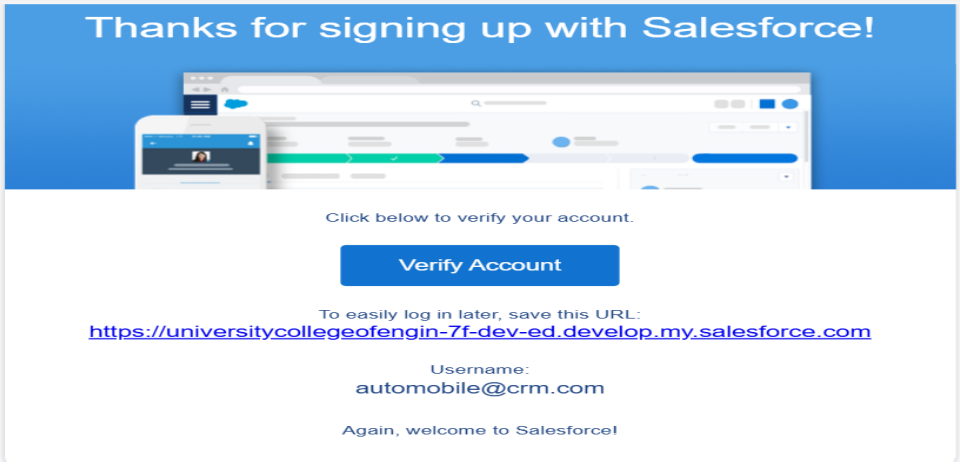

Fig:03

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

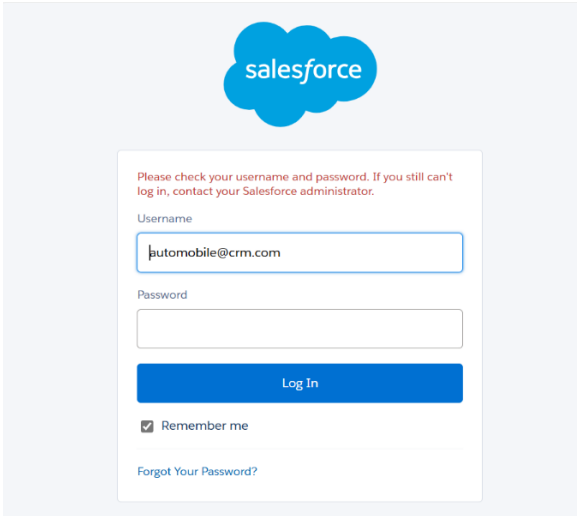

Fig:04

- Then you will redirect to your salesforce setup page.

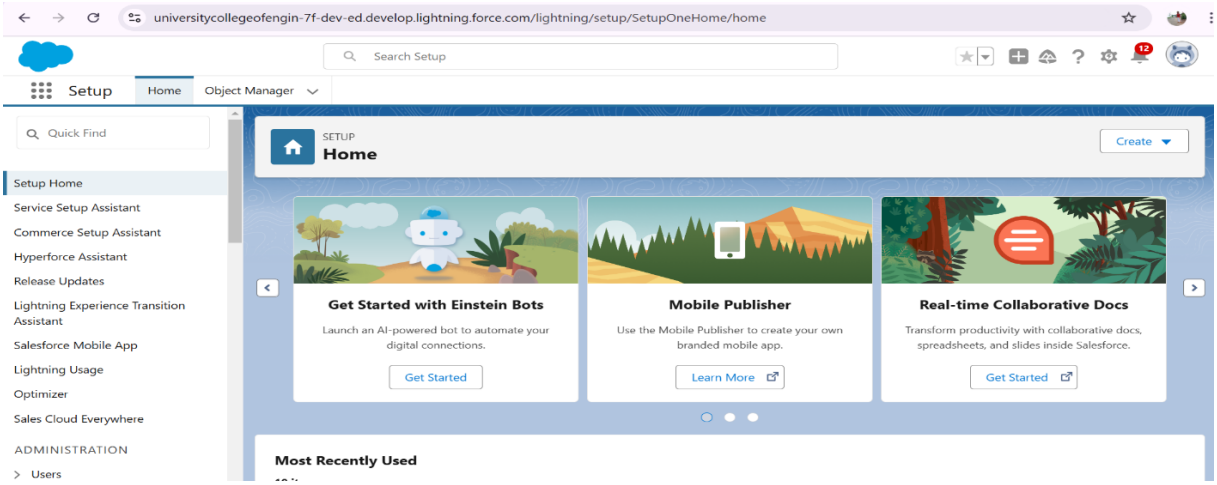

Fig:05

➢ **OBJECT**

Salesforce objects are database tables that permit you to store data that is specific to a organization.

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.

Create Invoice Object

Create Invoice object, just as we have created an Automobile Information Object using Make sure to Download the File into CSV Format.

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

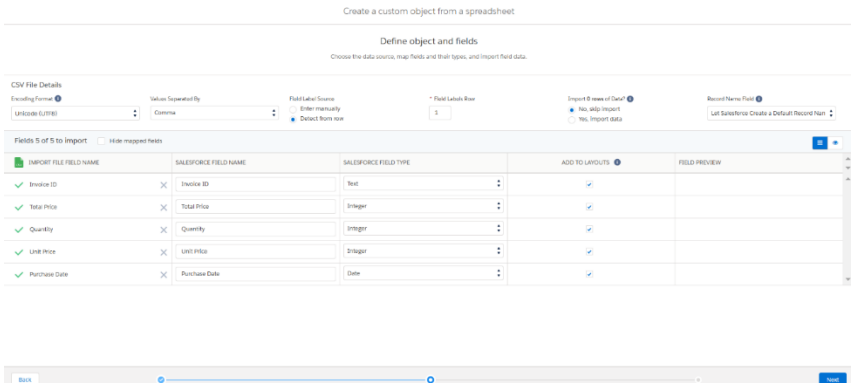

Fig:06

Create Automobile Object

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

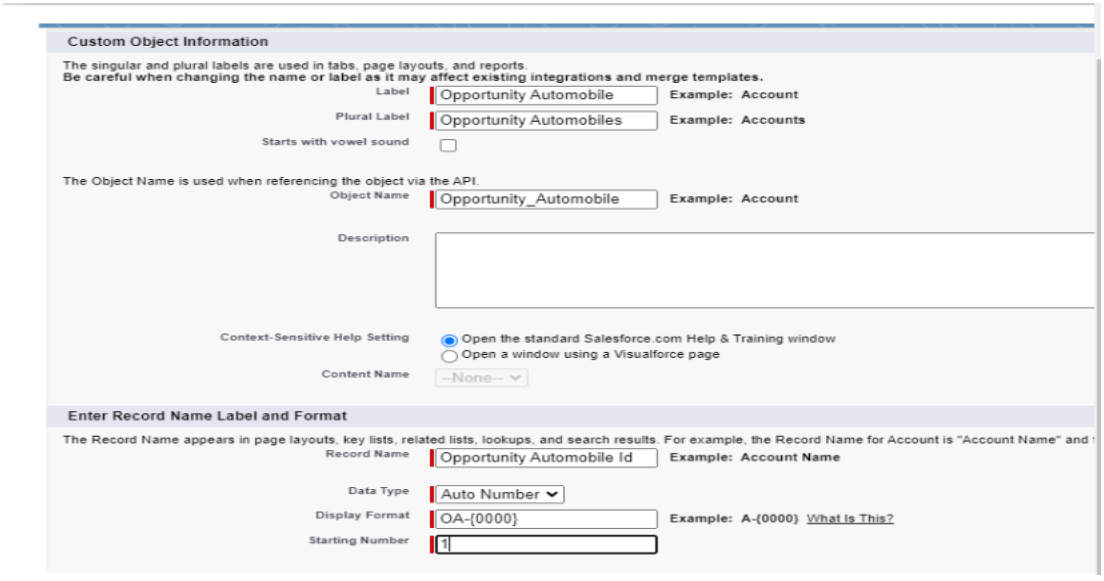

Fig:07

➢ 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.

Creating a Custom Tab

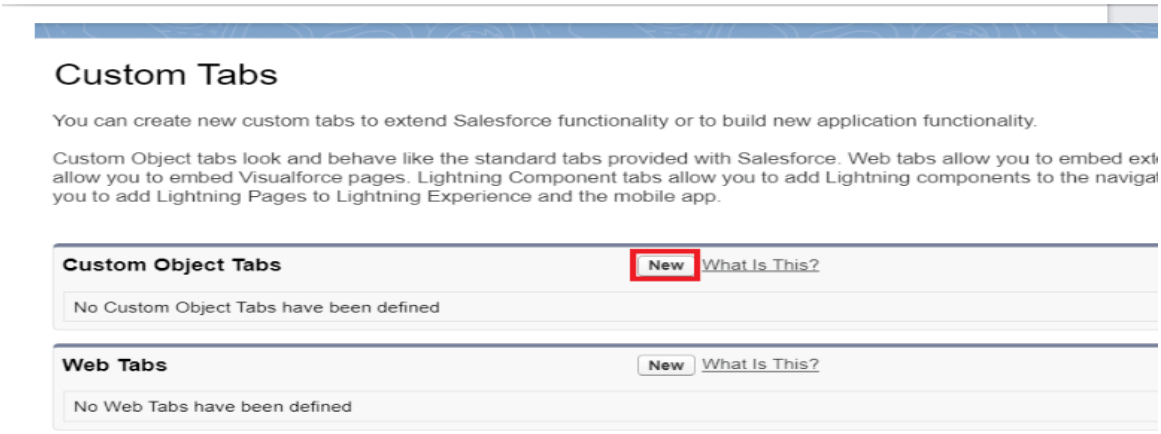

Fig:08

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.

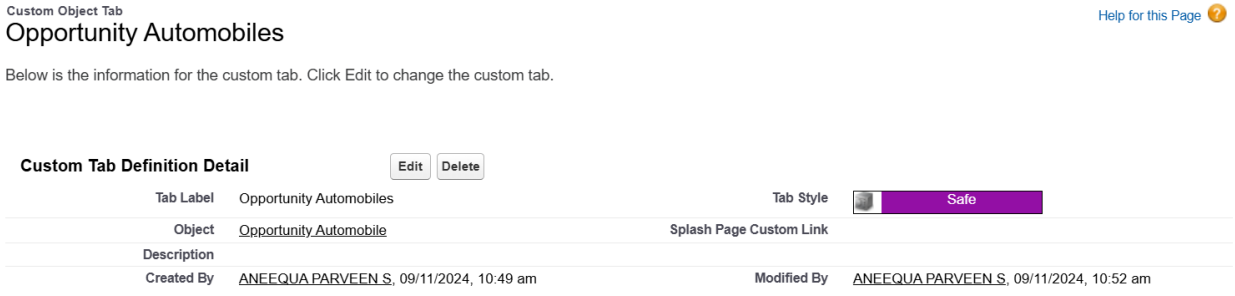

Fig:09

3.final Custom Tab

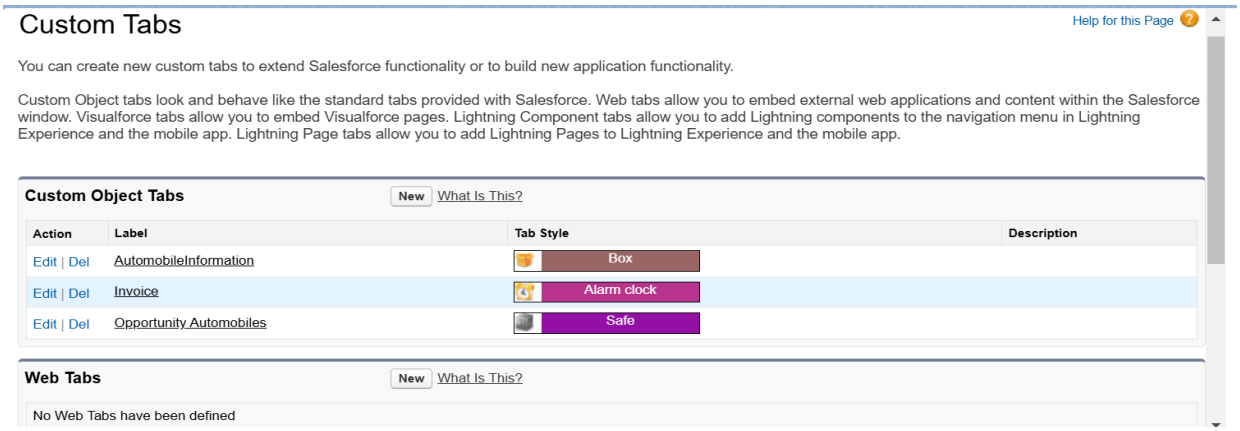

Fig:10

➢ 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.

Create a Lightning App

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

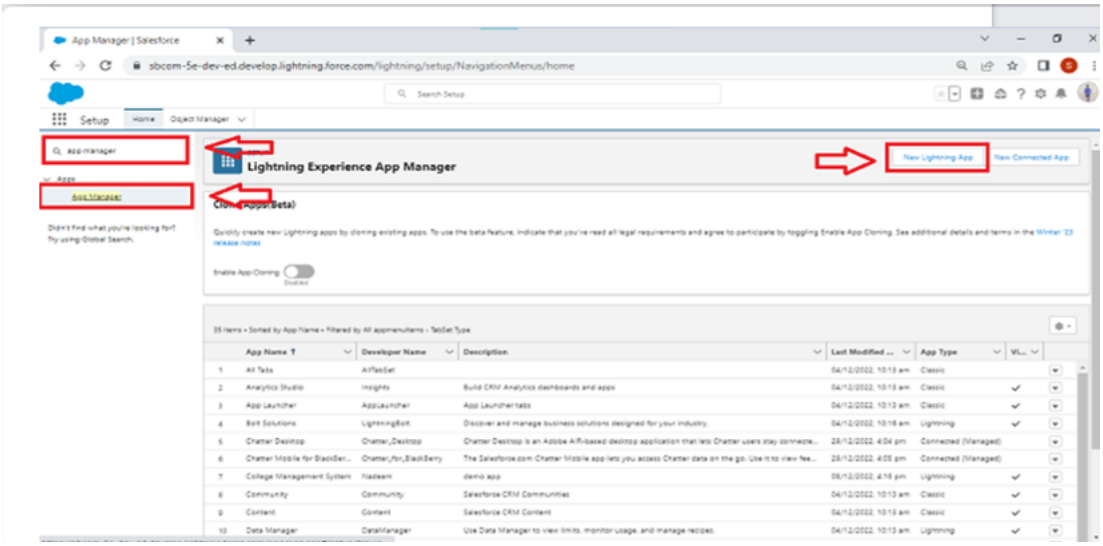

Fig:11

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

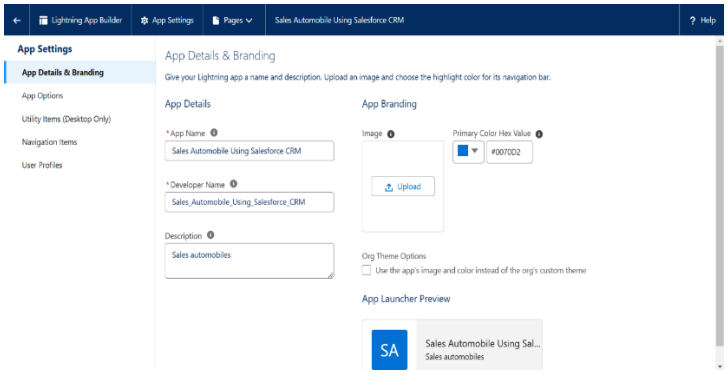

Fig:12

4.Add Navigation Items:

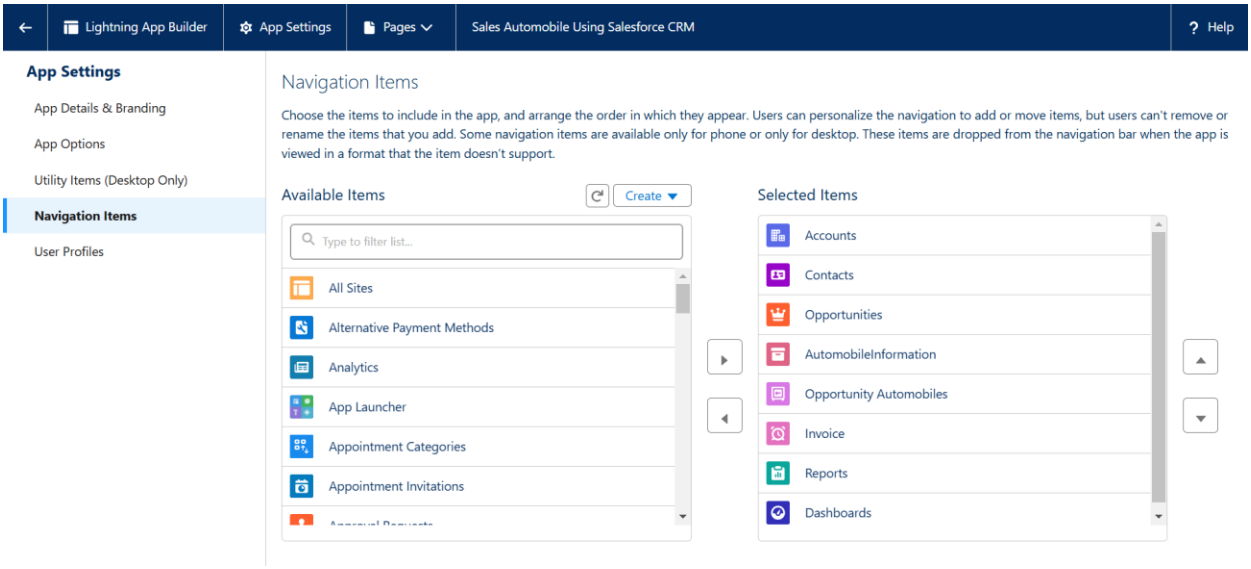

Fig:13

5.Search the items in the search bar(Account,Contact ,Opportunities,Automobile Information,Opportunity Automobile,Invoice, Reports, Dashboard) from the search bar and move 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.

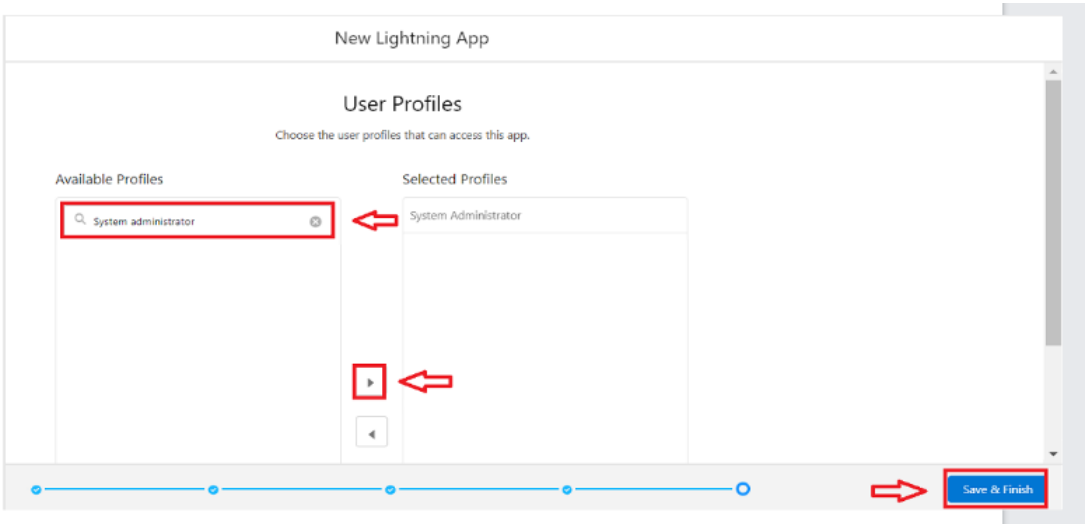

Fig:14

➢ 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.

➢ 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.

Edit the Page layout for Opportunity Object

- Go to Setup >> Click on Object Manager >> On the search bar, select Opportunity Layout. You can notice Page Layouts on the left panel
- Click on Page Layouts, Click on 'Opportunity Layouts'.
- In the Opportunity Detail Section, you can see various fields. Go on Account And Click on that Properties icon of Account name Field.
- check the Required box for Account name and click on Ok.
- Click on Save.

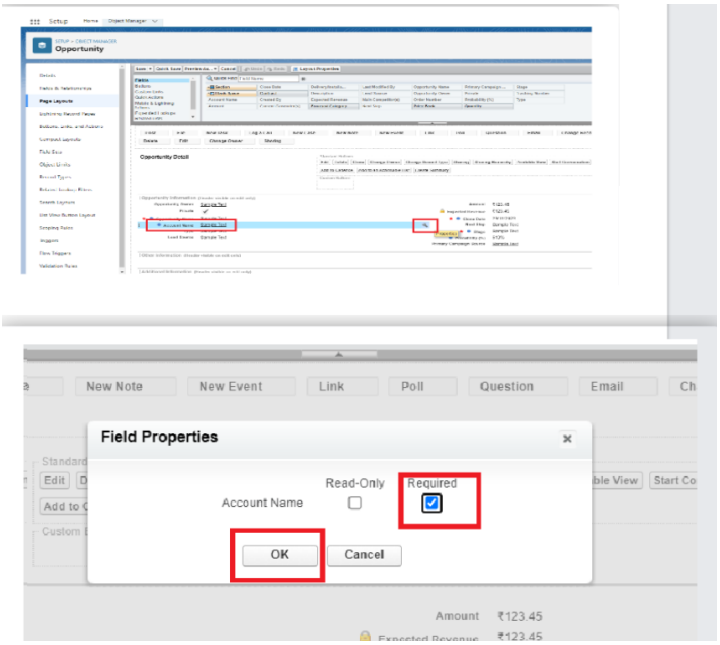

Fig:15

Edit the Page layout for Automobiles Information

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

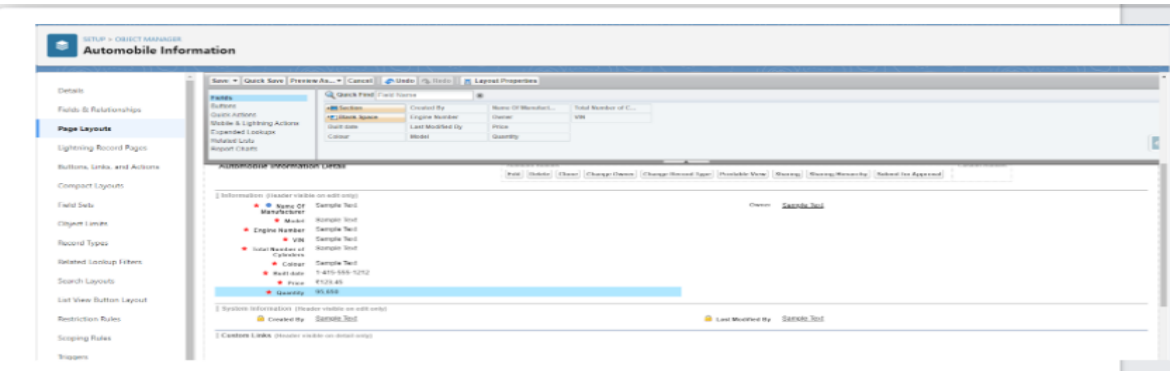

Fig:16

- Adjust the Fields as given below for A good looking view.

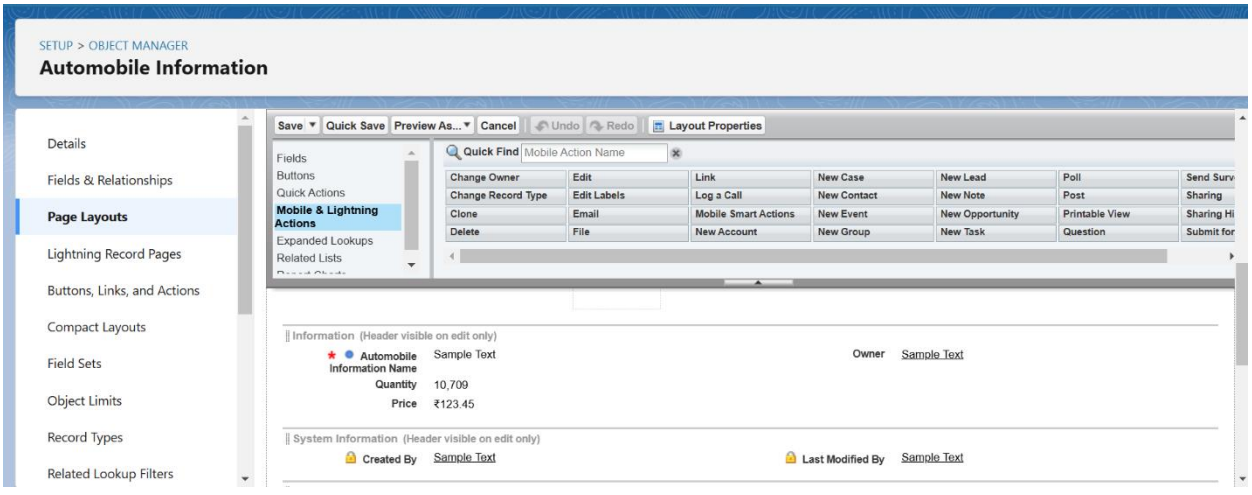

Fig:17

➢ 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

Before Trigger: This type of trigger in Salesforce is used either to update or validate the values of a record before they can be saved into the database. So, basically, the before trigger validates the record first and then saves it. Some criteria or code can be set to check data before it gets ready to be inserted into the database.After Trigger: This type of trigger in Salesforce is used to access the field values set by the system and affect any change in the record. In other words, the after trigger makes changes to the value from the data inserted in some other record.

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

- Login to the respective trailhead account and navigate to the gear icon in the top right corner.
- Click on the Developer console. Now you will see a new console window.
- In the toolbar, you can see FILE. Click on it and navigate to new and create New apex class.
- 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> lstAutomobileInformation = 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;
                lstAutomobileInformation.add(MapAutomobileInformation.get(AutoOpp.Automobile__c));
            }
        }
        If(!lstAutomobileInformation.IsEmpty()){
            update lstAutomobileInformation;
        }

    }
}
```

Fig:18

➢ LWC Component

Create Apex Class to Get Invoices

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.

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

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.

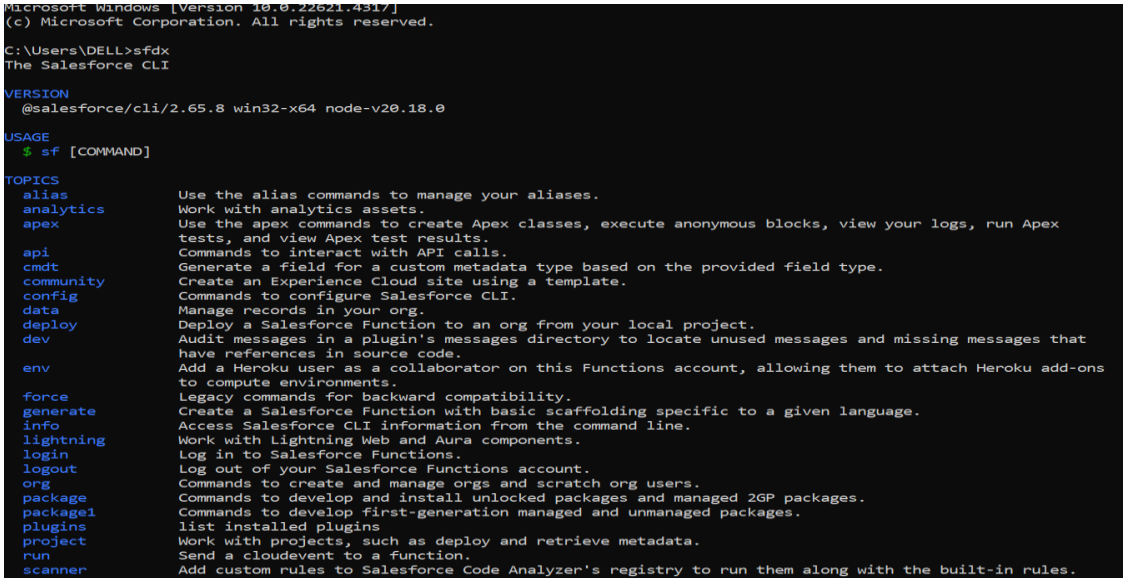

Fig:19

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.

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.

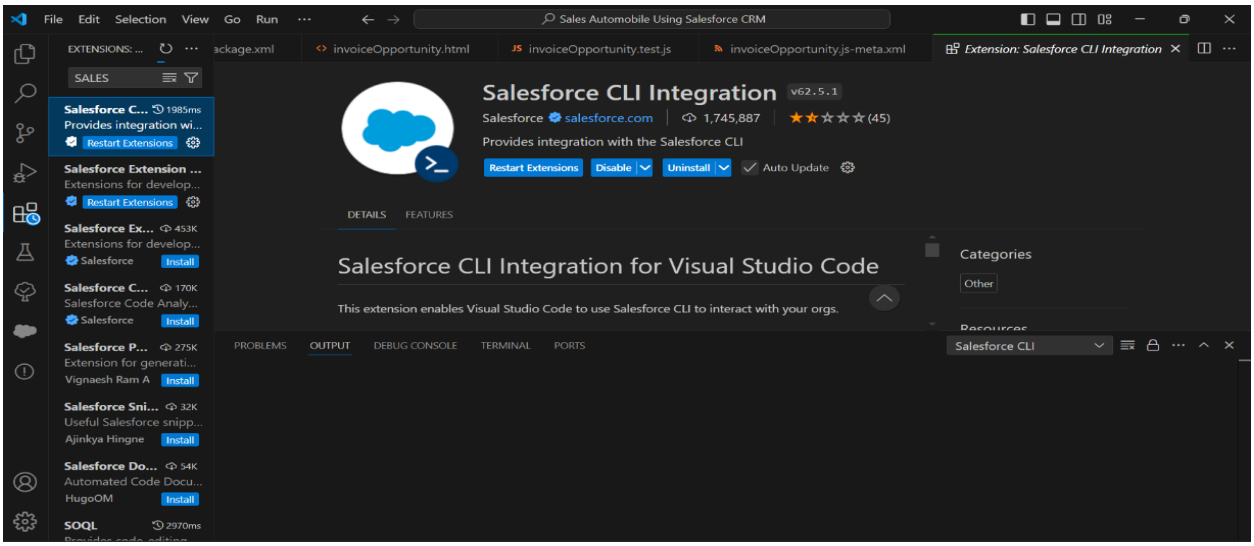

Fig:20

Add InvoiceOpportunity into Opportunity Record Page

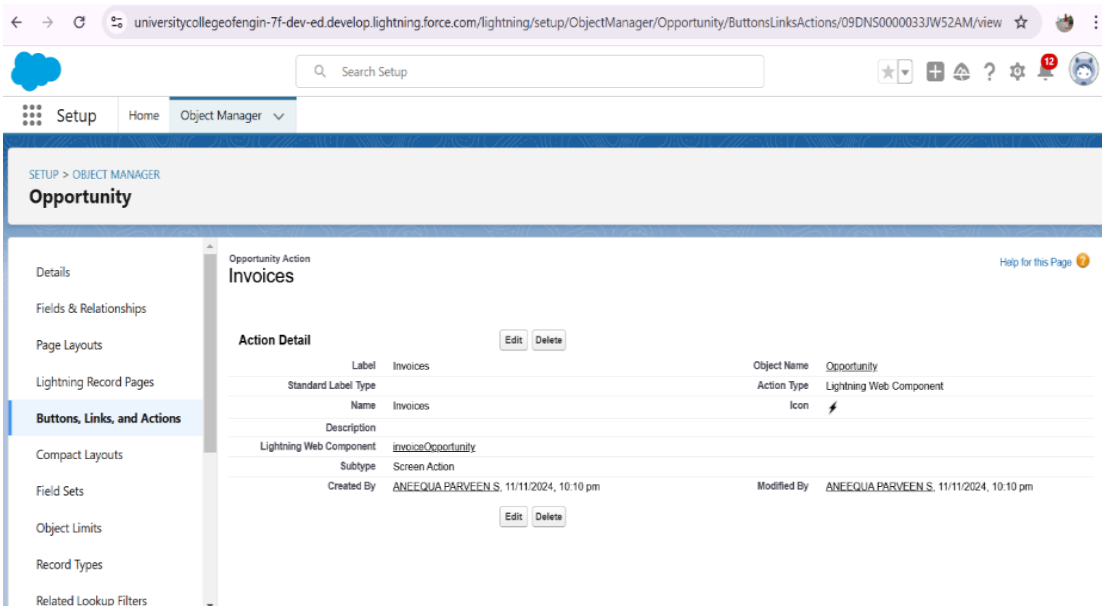

Fig:21

➢ 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
   }
}
```

Delete opportunity Schedule Class

Schedule the Apex class:

- Go to the Home page in your salesforce account.
- In the search bar, enter Apex and click on Apex Classes Click on Schedule Apex and enter the Job name

Job Name : DeleteOpportunitySchedule

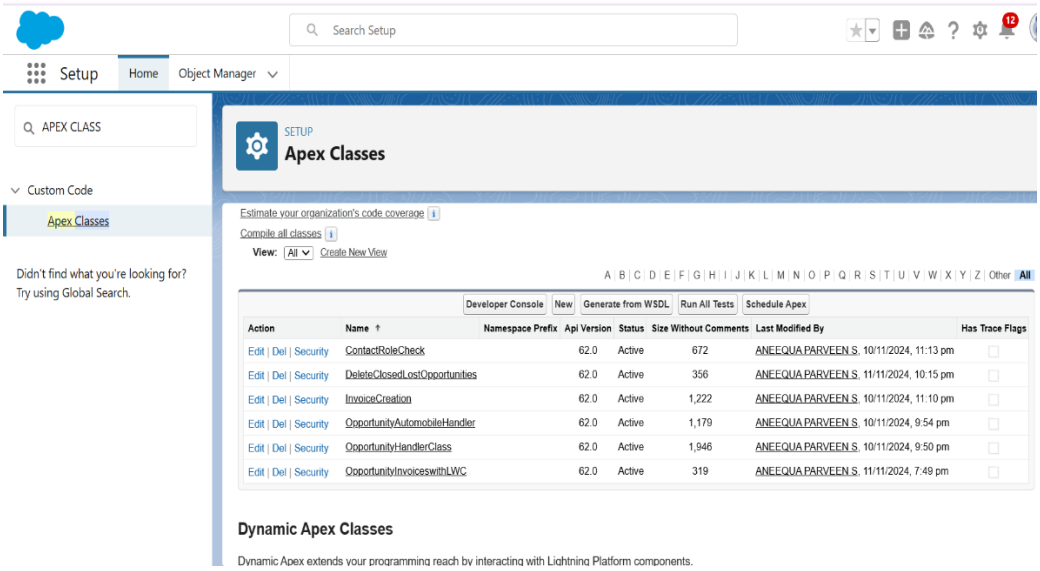

Fig:22

➢ 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.

Create Report on Opportunity

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

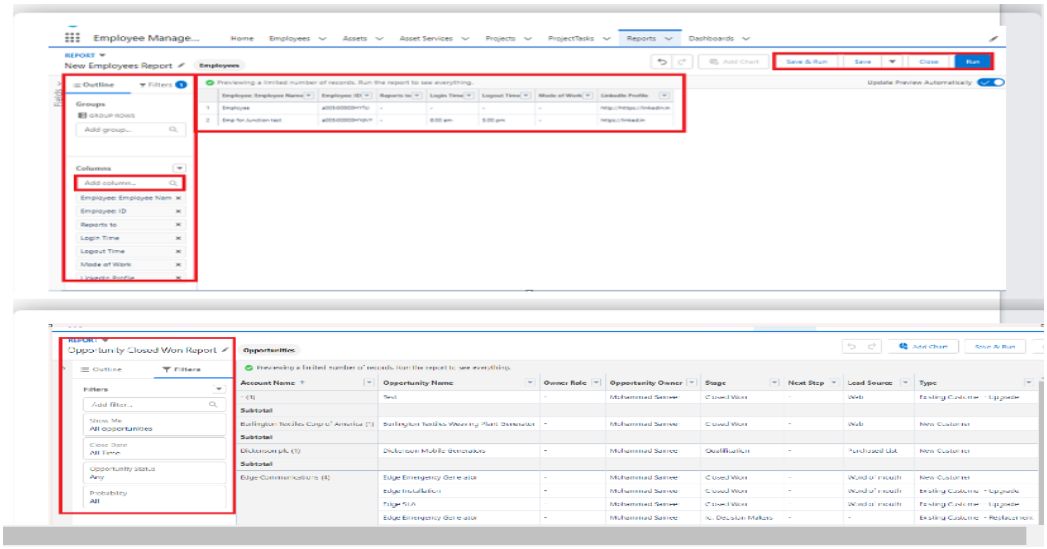

Fig:23

- Select report type from category or from report type panel or from search panel >> click on start report.
- Add fields from left pane as shown below

Create Report on Automobile Information

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

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

➢ 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.

Sales Dashboard

- Create Dashboard
- Select add component.
- Click Add then click on Save and then click on Done.

The Created Dashboard will look like this.

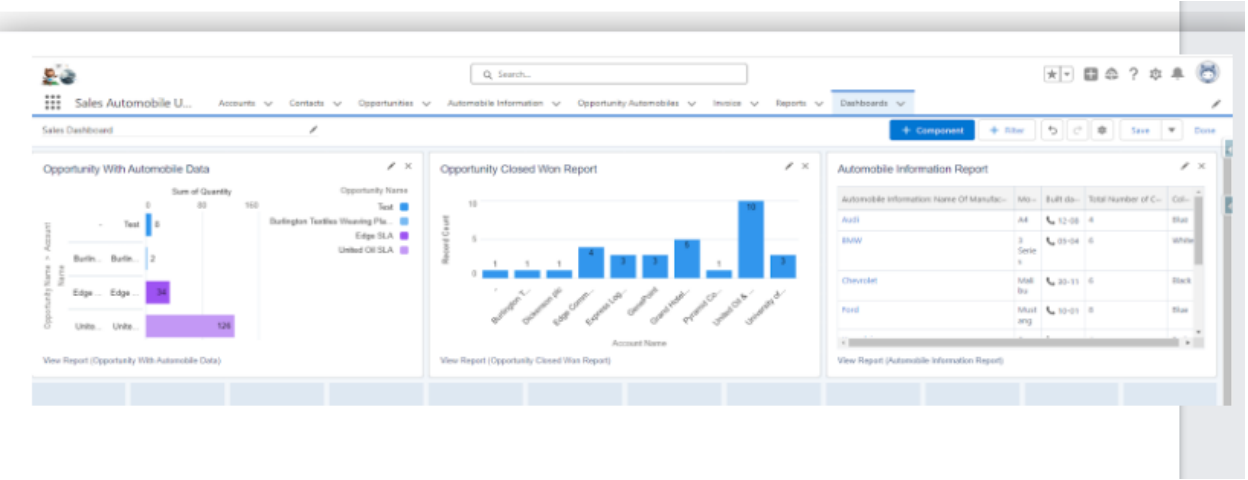

Fig:23

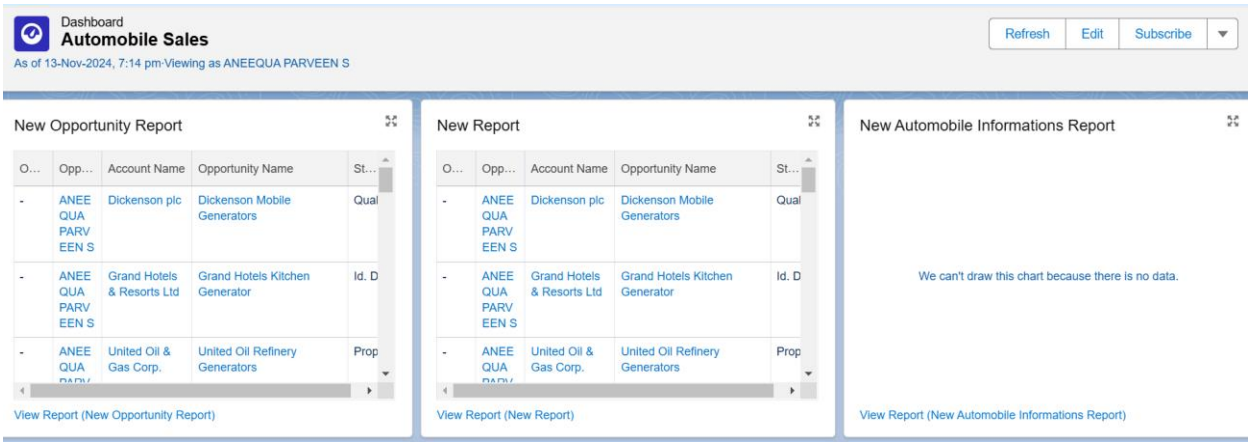

Fig:24

1.Detailed Steps to Solution Design:

- Consulted stakeholders to gather requirements specific to engineering workflows.
- Identified key CRM features to support customer interaction tracking, project status monitoring, and documentation management. 2. Salesforce Setup and Configuration:

2. Salesforce Setup and Configuration:

- Configured Salesforce instances with Sales Cloud.

- Created custom objects to accommodate data unique to engineering works (e.g., project specs, engineering documents).
- Set up roles and profiles to ensure data security and access control.

3. Data Modeling and Field Customization:

- Defined custom objects such as Projects, Clients, and Tasks.
- Added fields to capture essential data, including project start and end dates, budgets, materials used, etc.

4. Workflow Automation:

- Configured workflows to automate task assignments, lead conversions, and client follow-ups.
- Created rules that trigger email notifications for new leads or when projects reach critical milestones.

5. Integration with Engineering Tools:

- Integrated Salesforce with project management and engineering software to sync project updates in real time.
- Used APIs to allow seamless data transfer between Salesforce and other engineering tools.

6. Design of Dashboards and Reports:

- Developed dashboards to display key performance indicators, such as project progress, team efficiency, and client satisfaction.
- Created real-time reports for tracking the sales pipeline, revenue forecasts, and project health.

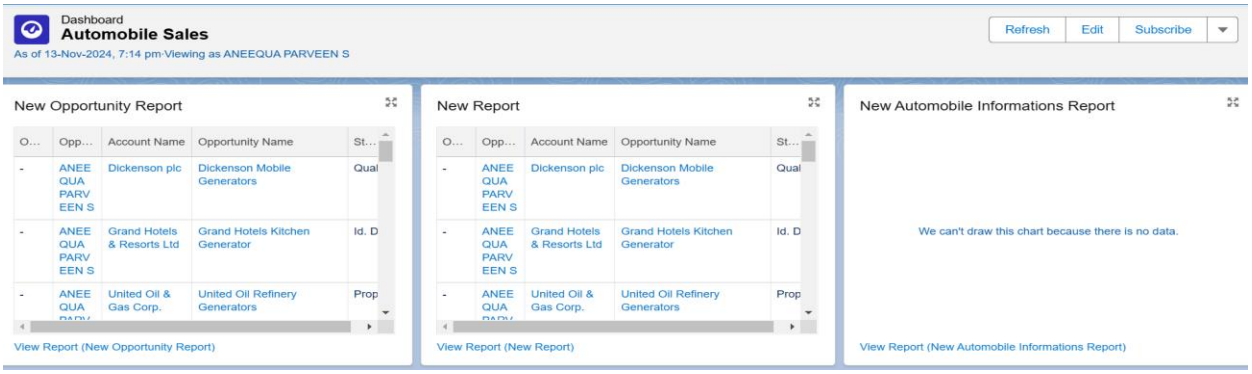

Fig:25

5. Testing and Validation

- Unit Testing: Validated each Salesforce module separately, including lead management, sales tracking, and customer follow-up features, ensuring individual functionalities work as expected.
- System Testing: Tested the entire CRM setup for seamless integration of all Salesforce features, including custom objects, workflows, and automation, to confirm they function cohesively.
- User Acceptance Testing (UAT): Conducted UAT with dealership stakeholders, confirming that the CRM solution met business requirements for automobile sales, lead tracking, and customer engagement.
- Load Testing: Tested system performance under various loads to ensure it could handle high traffic during peak sales periods, such as holiday promotions or sales events.

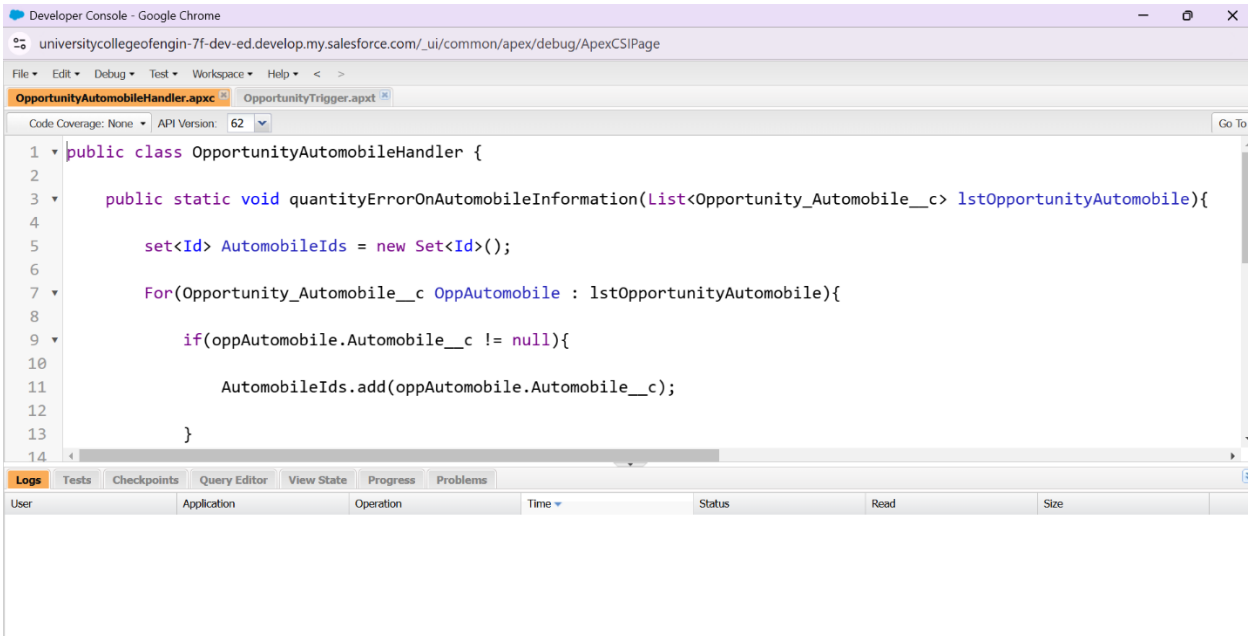

Fig:26

6. Key Scenarios Addressed by Salesforce in The Implementation Process

- **Lead Management**: Automated lead assignment to sales engineers, ensuring prompt follow-ups and reducing the chances of losing potential clients.
- **Project Lifecycle Tracking**: Allowed teams to track the status of each engineering project, from design and approval to implementation and delivery.
- **Client Communication and Documentation**: Centralized client communications and engineering documents within Salesforce for easy access and retrieval.
- **Forecasting and Budget Management:** Provided accurate forecasting of project costs and timelines to help engineers and managers make informed decisions.
- **Customer Feedback and Follow-Up**: Enabled tracking of customer satisfaction and automated reminders for post-project feedback collection.

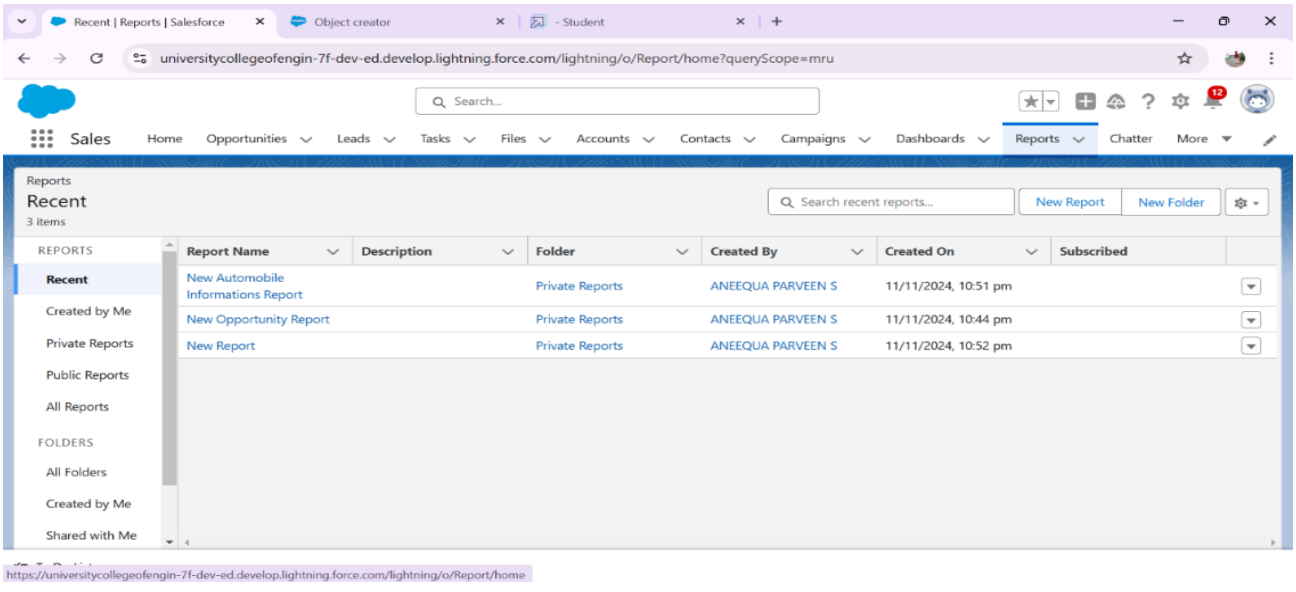

Fig:27

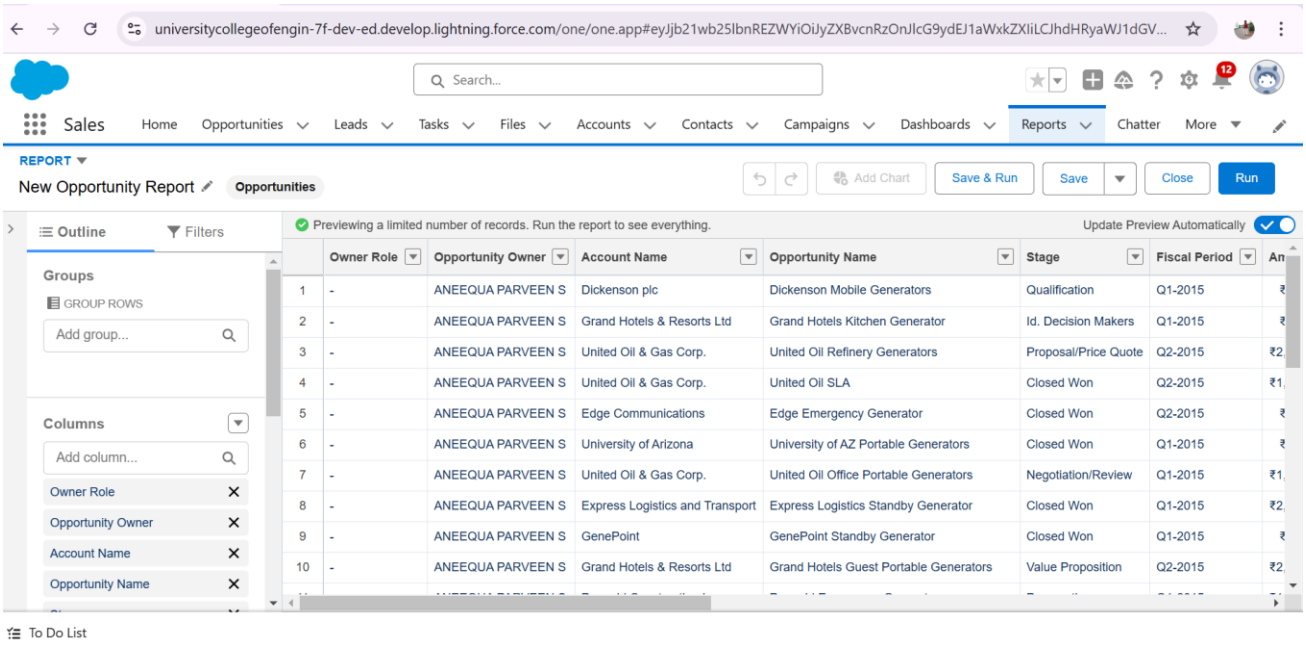

Fig:28

8. Conclusion

This CRM application for automobile sales successfully streamlined client relationship management, sales tracking, and customer communication. Salesforce's extensive CRM capabilities provide an efficient, data-driven solution for managing customer relationships and overseeing the automobile sales process.