

## **Field Service WorkOrder Optimization**

### **Project Description:**

The Field Service Work Order Optimization System streamlines operations for a company providing installations and repairs. Utilizing a robust database, the system efficiently matches work orders with skilled technicians based on technicians location, availability, and skills. The system employs a prioritization algorithm, focusing on assigning tasks to technician. Automated communication keeps technicians informed, while analytics offer insights for continuous improvement. Overall, this solution maximizes efficiency, reduces operational costs, and improves customer satisfaction in the dynamic realm of field service operations.

### **Short Description:**

Our field service work order optimization system seamlessly assigns tasks to technicians based on skills, location and availability. Leveraging a smart algorithm, it ensures efficient dispatching, reducing travel time and costs. Real-time updates and a user-friendly interface enhance productivity, delivering swift, precise and customer-centric service solutions.

### **Project Flow:**

Milestone-1: Creation of Developer Account

Milestone-2: Object Creation

Milestone-3: Tabs Creation

Milestone-4: Lightning App Creation

Milestone-5: Fields & Relationships

Milestone-6: Profiles

Milestone-7: Users

Milestone-8: Apex Trigger

Milestone-9: Reports & Dashboards

## **Milestone 1- Creation of Developer Account**

### **What is Salesforce?**

Salesforce is a cloud-based software company that provides businesses with tools that help them find more prospects, close more deals, and provide a higher level of service to their customers.

Salesforce Inc is a famous American cloud-based software company that provides CRM services. Salesforce is a popular CRM tool for support, sales, and marketing teams worldwide.

Salesforce services allow businesses to use cloud technology to better connect with partners, customers, and potential customers. Using the Salesforce CRM, companies can track customer activity, market to customers, and many more services.

A CRM platform helps you go deeper with all your metrics and data; you could also set up a dashboard that showcases your data visually. In addition to this, you can also have personalized outreach with automation. Another significant benefit is that a CRM platform can also improve customer service's ability to help customers or a sales team's outreach efforts.

### **Task 1: Developer Account**

Creating a developer org in salesforce.

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

**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\*  
Your first name

Last Name\*  
Your last name

Email\*  
Your email address

Role\*  
Your job role

Company\*  
Company Name

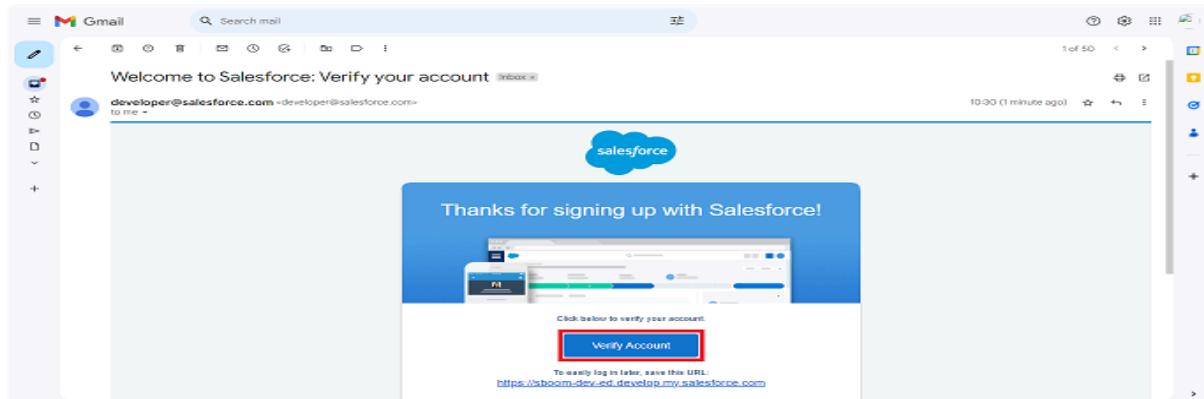
1. First name & Last name
2. Email
3. Role : Developer
4. Company : College Name
5. County : India
6. Postal Code : pin code
7. Username : should be a combination of your name and company

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.

## Task 2: 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.



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



## Change Your Password

Enter a new password for lead@sb.oom.  
Make sure to include at least:

- 8 characters
- 1 letter
- 1 number

\* New Password  
 Good

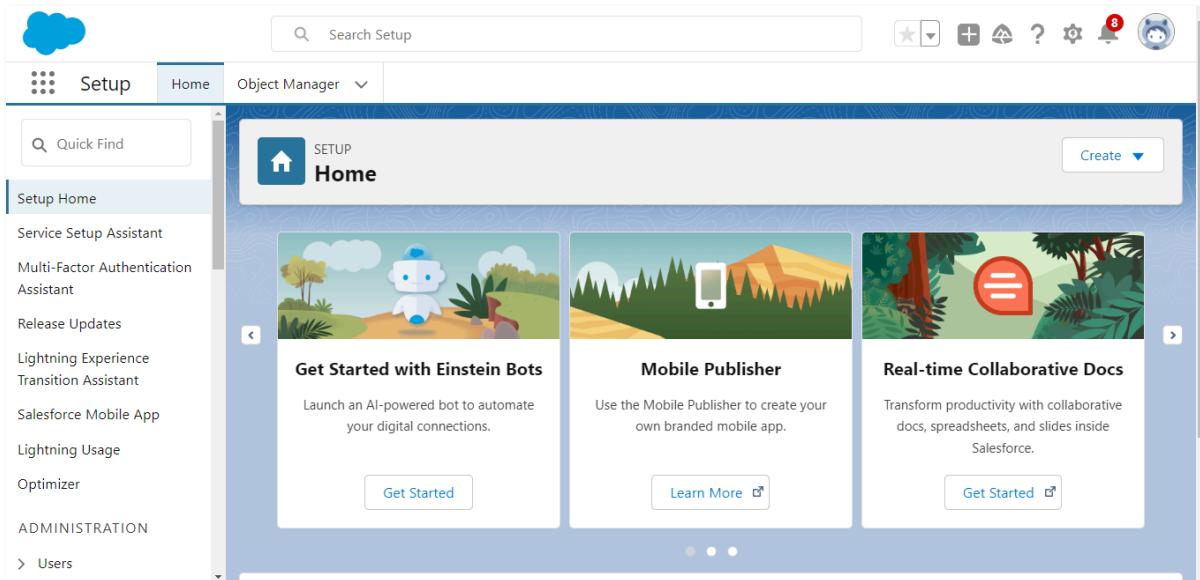
\* Confirm New Password  
 Match

Security Question  
 ▾ In what city were you born?

\* Answer

**Change Password**

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



The screenshot shows the Salesforce Setup Home page. The left sidebar includes links for Setup Home, Service Setup Assistant, Multi-Factor Authentication Assistant, Release Updates, Lightning Experience Transition Assistant, Salesforce Mobile App, Lightning Usage, Optimizer, Administration, and Users. The main content area features three cards: "Get Started with Einstein Bots" (Launch an AI-powered bot to automate your digital connections), "Mobile Publisher" (Use the Mobile Publisher to create your own branded mobile app), and "Real-time Collaborative Docs" (Transform productivity with collaborative docs, spreadsheets, and slides inside Salesforce). A "Create" button is located in the top right corner of the main content area.

## Milestone 2- Object Creation

Objects are database tables that allow you to store data specific to an organization.

**Salesforce objects are of two types:**

- **Standard Objects:** Standard objects are the kind of objects that are provided by [salesforce.com](https://salesforce.com) such as users, contracts, reports, dashboards, etc.
- **Custom Objects:** Custom objects are created by users. They collect information that is unique and essential to their organization. They are the heart of any application and provide a structure for sharing data.

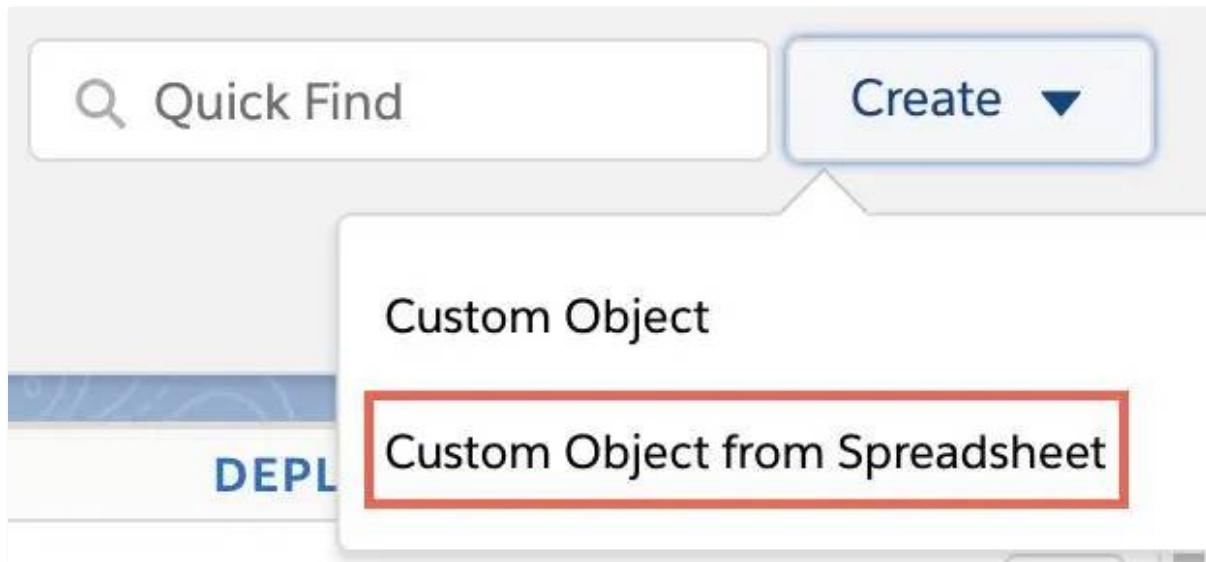
**Task 1: Technician Object**

Download and open [this spreadsheet](#), edit the email column (provide your email for at least one or two records) and save it as Technician.csv.

1. Log into your salesforce account, click on Gear icon, then select Setup.
2. Click the Object Manager tab.



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



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

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

CSV File Details

Encoding Format: Unicode (UTF8) Values Separated By: Comma Field Label Source: Detect from row Field Labels Row: 1 Import 5 rows of Data? Yes, import data

Record Name Field: TechnicianID

IMPORT FILE FIELD NAME	SALESFORCE FIELD NAME	SALESFORCE FIELD TYPE	ADD TO LAYOUTS	FIELD PREVIEW
✓ Technician ID	Technician ID	Text	<input checked="" type="checkbox"/>	T-0001
✓ Name	Name	Text	<input checked="" type="checkbox"/>	Raghu
✓ Phone	Phone	Phone	<input checked="" type="checkbox"/>	7892341560
✓ Email	Email	Email	<input checked="" type="checkbox"/>	example@gmail.com
✓ Location	Location	Picklist	<input checked="" type="checkbox"/>	Hyderabad
✓ Availability	Availability	Picklist	<input checked="" type="checkbox"/>	Available
✓ Skills	Skills	Picklist	<input checked="" type="checkbox"/>	Machine Installation

Fields 7 of 7 to import Hide mapped fields

Back Next

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

## Task 2: WorkOrder Object

Create WorkOrder object, just as we have created Technician Object using [this spreadsheet](#):

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

CSV File Details

Encoding Format: Unicode (UTF8) Values Separated By: Comma Field Label Source: Detect from row Field Labels Row: 1 Import 0 rows of Data? No, skip import

Record Name Field: WorkOrder ID

IMPORT FILE FIELD NAME	SALESFORCE FIELD NAME	SALESFORCE FIELD TYPE	ADD TO LAYOUTS	FIELD PREVIEW
✓ WorkOrder ID	WorkOrder ID	Text	<input checked="" type="checkbox"/>	
✓ Email	Email	Email	<input checked="" type="checkbox"/>	
✓ Service Type	Service Type	Picklist	<input checked="" type="checkbox"/>	
✓ Description	Description	Text Area (Long)	<input checked="" type="checkbox"/>	
✓ Location	Location	Picklist	<input checked="" type="checkbox"/>	
✓ Priority	Priority	Picklist	<input checked="" type="checkbox"/>	
✓ Status	Status	Picklist	<input checked="" type="checkbox"/>	

Fields 7 of 7 to import Hide mapped fields

Back Next

## Task 3: Assignment Object

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 with the 'Object Manager' tab selected. A red arrow points to the 'Object Manager' tab. Another red arrow points to the 'Custom Object' button in the top right corner of the main content area.

1. Enter the label name --> Assignment
2. Plural label name --> Assignments

The screenshot shows the 'Custom Object Definition Edit' page. The 'Label' field is highlighted with a red box and has an example 'Example: Account'. The 'Plural Label' field is also highlighted with a red box and has an example 'Example: Accounts'. A red arrow points to the 'Save' button.

3. Enter Record Name Label and Format
  - Record Name --> Assignment ID
  - Data Type --> Auto Number
  - Display Format --> A-{0000}
  - Starting Number --> 1

The screenshot shows the 'Enter Record Name Label and Format' page. The 'Record Name' field is highlighted with a red box and contains 'Assignment ID'. The 'Data Type' field is highlighted with a red box and contains 'Auto Number'. The 'Display Format' field is highlighted with a red box and contains 'A-{0000}'. The 'Starting Number' field is highlighted with a red box and contains '1'. Examples are provided for each field: 'Example: Account Name' for Record Name, 'Example: A-{0000} What Is This?' for Display Format.

4. Click on Allow reports,

Allow search --> Save.

### Milestone 3- Tabs Creation

Tabs are the user interface elements that provide access to objects, functions, or data. They represent the front end of your Salesforce experience. A tab is like a user interface that is used to build records for objects and to view the records in the objects.

#### Task 1: Creating a Custom Tab

To create a Tab:(Assignment)

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(Assignment) --> Select any tab style --> Next (Add to profiles page) keep it as default --> Next (Add to Custom App) keep it as default --> Save.

## New Custom Object Tab

**Step 1. Enter the Details**

Choose the custom object for this new custom tab. Fill in other details.

Select an existing custom object or [create a new custom object now](#).

Object: Assignment

Tab Style: --None-- Assignment

(Optional) Choose a Home Page Custom Link to show as a splash page the first time your users click on this tab.

Splash Page Custom Link: --None--

Note: Tabs for WorkOrder & Technician objects do get created automatically. We do not need to create tabs for those objects.

## Milestone 4- Lightning App Creation

An app is a collection of items that work together to serve a particular function. In Lightning Experience, Lightning apps give your 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.

### Task 1: Create a lightning app

To create a lightning app page:

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

The screenshot shows the Salesforce App Manager interface. At the top, there's a search bar with 'app manager' typed in. Below it, a sidebar has 'Clone (App/Beta)' highlighted with a red box. In the top right corner, there are two buttons: 'New Lightning App' and 'New Connected App', both highlighted with red boxes.

- Fill the app name in app details and branding as follow

App Name : Field Service WorkOrder Optimization

Developer Name : this will auto populated

Description : Give a meaningful description

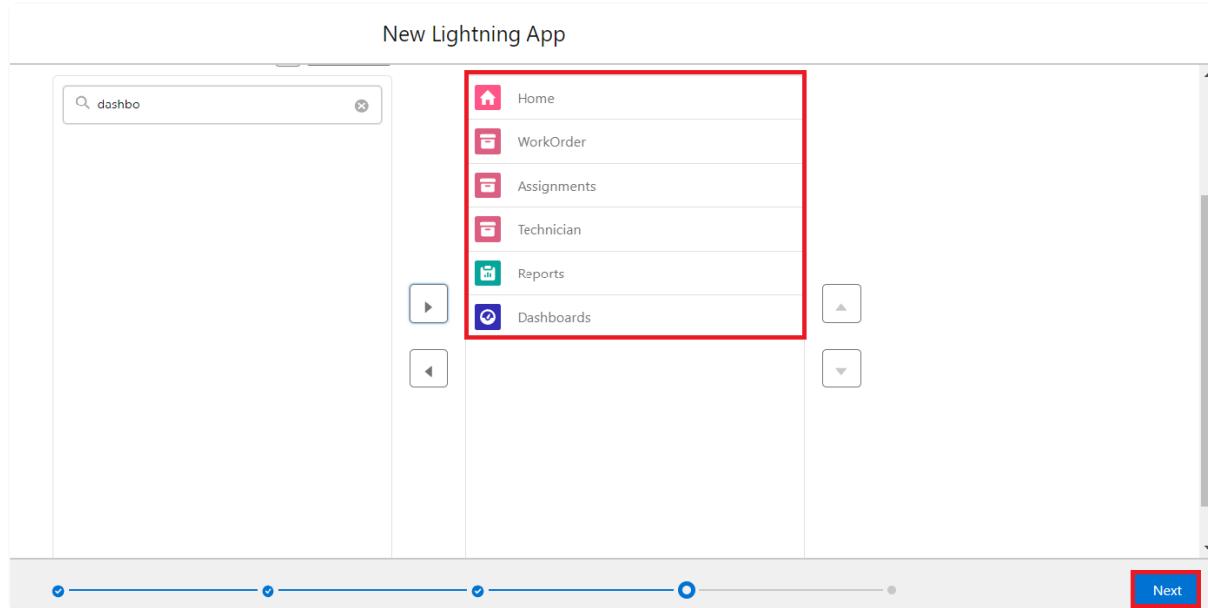
Image : optional (if you want to give any image you can otherwise not mandatory)

Primary color hex value : keep this default

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

The screenshot shows the 'New Lightning App' configuration page. Under 'App Details & Branding', there is a section for 'App Details' with fields for 'App Name' (highlighted with a red box), 'Developer Name' (with placeholder 'Enter a developer name...'), and 'Description' (with placeholder 'Enter a description...'). To the right is the 'App Branding' section with fields for 'Image' (with a 'Upload' button) and 'Primary Color Hex Value' (#007002). At the bottom right of the page, there is a large red arrow pointing to the 'Next' button.

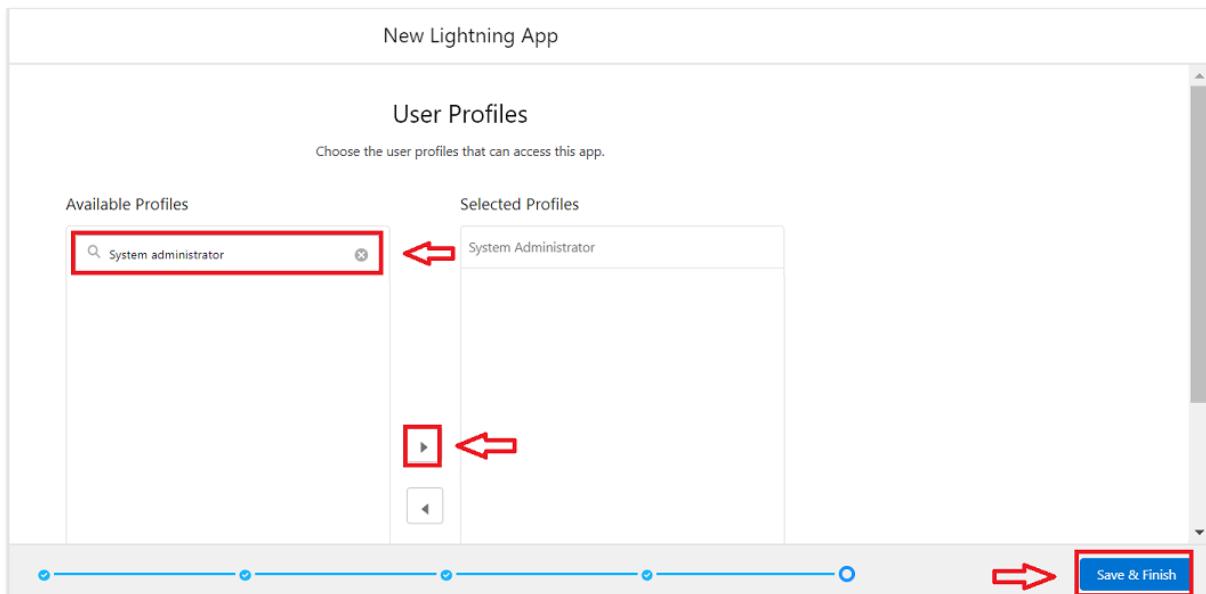
- To Add Navigation Items:



Search the items in the search bar(Home, WorkOrder, Technician, Assignment, 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.

##### 5. To Add User Profiles:



Search profiles (System administrator) in the search bar --> click on the arrow button

--> save & finish.

## Milestone 5- Fields & Relationships

### Task 1: Creating Lookup Field in Assignment Object

To create fields in an object:

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

The screenshot shows the Salesforce Object Manager. At the top, there's a navigation bar with 'Setup', 'Home', and 'Object Manager'. A red box highlights 'Object Manager' with a red arrow pointing to it. In the search bar at the top right, 'assignment' is typed, and a red box highlights the search term with a red arrow pointing to it. Below the search bar is a table listing objects. The first row has 'Assignment' highlighted with a red box and a red arrow pointing to it. The table columns are labeled: LABEL, API NAME, TYPE, DESCRIPTION, LAST MODIFIED, and DEPLOYED. The 'Assignment' row shows 'Assignment\_c' as the API name, 'Custom Object' as the type, and '20/11/2023' as the last modified date.

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

The screenshot shows the 'Assignment' object's Fields & Relationships page. At the top, there's a navigation bar with 'Setup', 'Home', and 'Object Manager'. Below it, the object name 'Assignment' is shown. On the left, there's a sidebar with options like 'Details', 'Fields & Relationships' (which is highlighted with a red box and a red arrow), 'Page Layouts', 'Lightning Record Pages', 'Buttons, Links, and Actions', 'Compact Layouts', and 'Field Sets'. The main area is titled 'Fields & Relationships' and shows a table of existing fields. A red box highlights the 'New' button at the top right of the table area, with a red arrow pointing to it. The table columns are: FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED.

3. Select Data type as “Lookup”.

The screenshot shows the 'Step 1. Choose the field type' page. At the top, there's a header with 'Step 1' and 'Next' (highlighted with a red box and a red arrow). Below the header, there's a text input field with placeholder text 'Specify the type of information that the custom field will contain.' Underneath, there's a section titled 'Data Type' with several radio button options. The 'None Selected' option is selected. The 'Lookup Relationship' option is highlighted with a red box and a red arrow, with a tooltip explaining it creates a relationship field. The other options are: 'Auto Number', 'Formula', 'Roll-Up Summary', and 'Master-Detail Relationship'.

4. Click on Next

5. For field label related to: select “WorkOrder” object and click Next.

Note: Do not select other standard object with the same name for sake of ease copy the above and paste it.

Assignment  
New Relationship

Help for this Page 

**Step 2. Choose the related object**

Select the other object to which this object is related.

Related To  

**Step 2**

[Previous](#) [Next](#) [Cancel](#)

[Previous](#)  [Next](#) [Cancel](#)

- Give Field Label as "WorkOrder ID" and click Next.

Assignment  
New Relationship

Help for this Page 

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

**Step 3 of 6**

Field Label  

Field Name  

Description

Help Text

[Previous](#)  [Next](#) [Cancel](#)

Next --> Next --> Save & New

### Task 2: Manage your picklist values

- From the setup page go to object manager
- Search and Select WorkOrder object.
- Go to fields & relationship, select Location field, scroll down to values and click "New".

Values					
		<a href="#">New</a> <a href="#">Reorder</a> <a href="#">Replace</a> <a href="#">Printable View</a> <a href="#">Chart Colors</a> 			
Action	Values	API Name	Default	Chart Colors	Modified By
<input type="checkbox"/>	<a href="#">Edit</a>   <a href="#">Del</a>   <a href="#">Deactivate</a>	Pune	<input type="checkbox"/>	Assigned dynamically	<a href="#">demo project</a> 22/11/2023, 9:53 am
<input type="checkbox"/>	<a href="#">Edit</a>   <a href="#">Del</a>   <a href="#">Deactivate</a>	Hyderabad	<input type="checkbox"/>	Assigned dynamically	<a href="#">demo project</a> 22/11/2023, 9:53 am

- Add the below values:
- Nasik
- Warangal
- Nanded
- Click Save.

#### Add Picklist Values

### Location

Add one or more picklist values below. Each value should be on its own line and it is used for both a value's label and API name.

If a value matches an inactive value's API name, that value is reactivated with its previous label.

If a value matches an inactive value's label but not the API name, a new value is created.

The screenshot shows a user interface for adding picklist values. At the top, there is a header 'Add Picklist Values' and a section title 'Location'. Below this, a text area contains three lines of text: 'Nasik', 'Warangal', and 'Nanded'. A red rectangular box is drawn around this text area. At the bottom right of the interface, there are two buttons: 'Save' and 'Cancel'. A second red rectangular box is drawn around the 'Save' button.

Add following values to the respective fields in WorkOrder object:

Field	Values
Priority	High
Service Type	Hardware repair Troubleshoot/Debugging Lane-Management

### Task 3: Creating Formula Field in WorkOrder Object

1. Repeat step 1 and 2 mentioned in activity 1
2. Select Data type as "Formula" and click Next.
3. Give Field Label and Field Name as "Date" and select formula return type as "Date" and click next.

Step 2. Choose output type Step 2 of 5

Field Label  Field Name  Previous Next Cancel

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

**Formula Return Type**

None Selected Select one of the data types below.

Checkbox Calculate a boolean value  
Example: `TODAY() > CloseDate`

Currency Calculate a dollar or other currency amount and automatically format the field as a currency amount  
Example: `Gross Margin = Amount - Cost_c`

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

Date/Time Calculate a date/time, for example, by adding a number of hours or days to another date/time.  
Example: `Next = NOW() + 1`

Number Calculate a numeric value  
Example: `Fahrenheit = 1.8 * Celsius_c + 32`

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

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

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

4. Under Advanced Formula write down the formula and click “Check Syntax”

Formula: CreatedDate

Simple Formula Advanced Formula

Insert Field

Date (Date) =

**Check Syntax** No syntax errors in merge fields or functions. (Compiled size: 20 characters)

5. Next--> Next--> Save.

#### Task 4: Creating Remaining fields for the respective objects

SI No	Object Name	Field	
		Field Name	Datatype
1	Assignment	<ul style="list-style-type: none"> <li>• <u>I</u> Technician ID</li> <li>• Assignment Date</li> <li>• Completion Date</li> </ul>	Lookup(Technician) Formula: return type : Date <code>(WorkOrder_ID__r.Date__c)</code> Formula: return type : Date <code>IF(ISPICKVAL(WorkOrder_ID__r.Status__c , 'Resolved'),</code>

#### Milestone 6- Profiles

Profile defines what an user is able to do or see in the Salesforce Org

## Technician Profile

1. Go to setup --> type profiles in quick find box --> click on profiles --> click on new profile.

The screenshot shows the 'Profiles' page in Salesforce setup. At the top left is a blue 'SETUP' button with a gear icon. Next to it is a blue 'Profiles' button with a person icon. Below these are tabs for 'All Profiles' (selected), 'Edit', 'Delete', and 'Create New View'. A red box highlights the 'New Profile' button at the top left of the main content area. The main area lists profiles with columns for 'Action', 'Profile Name', 'User License', and 'Custom'. The 'Profile Name' column lists 'Analytics Cloud Integration User', 'Analytics Cloud Security User', 'Authenticated Website', 'Authenticated Website', and 'External Apps Login'. The 'User License' column lists 'Analytics Cloud Integration User', 'Analytics Cloud Integration User', 'Authenticated Website', 'Authenticated Website', and 'Custom'. The 'Custom' column has checkboxes for each row, with the last one checked.

2. Select 'Standard Platform User' for existing profile and give 'Technician' for Profile Name and click on Save.

## Clone Profile

Enter the name of the new profile.

You must select an existing profile to clone from.

Existing Profile	Standard Platform User
User License	Salesforce Platform
Profile Name	[Empty Input Field]

Save Cancel

3. While still on the profile page, then click Edit.
4. Scroll down to Custom Object Permissions and Give Read only access permissions for Technician, WorkOrder and Assignment objects and field access permission as shown below:

Custom Object Permissions						
	Basic Access			Data Administration		
	Read	Create	Edit	Delete	View All 	Modify All 
Assets	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Asset Services	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Assignments	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Billings	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bookings	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Candidates	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Child object	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Crews	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Customer Orders	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Employees	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Employment Websites	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Flights	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Items	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Jewel Customers	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Job Applications	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Lead Scoring Rules	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Custom Object Permissions						
	Basic Access			Data Administration		
	Read	Create	Edit	Delete	View All 	Modify All 
Leaves	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Parent object 1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Parent object 2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Passengers	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Positions	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Prices	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Projects	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ProjectTasks	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Reviews	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Sessions	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Students	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
StudentSessions	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technician	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Trainers	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
WorkOrder	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

5. Scroll down and Click on Save.
6. Now from the profile detail page scroll down to custom field level security click on view next to WorkOrder object.

Custom Field-Level Security	
Asset <a href="#">[ View ]</a>	Lead Scoring Rule <a href="#">[ View ]</a>
Asset Service <a href="#">[ View ]</a>	Leave <a href="#">[ View ]</a>
Assignment <a href="#">[ View ]</a>	Parent object 1 <a href="#">[ View ]</a>
Billing <a href="#">[ View ]</a>	Parent object 2 <a href="#">[ View ]</a>
Booking <a href="#">[ View ]</a>	Passenger <a href="#">[ View ]</a>
Candidate <a href="#">[ View ]</a>	Position <a href="#">[ View ]</a>
Child object <a href="#">[ View ]</a>	Price <a href="#">[ View ]</a>
Crew <a href="#">[ View ]</a>	Project <a href="#">[ View ]</a>
Customer Order <a href="#">[ View ]</a>	ProjectTask <a href="#">[ View ]</a>
Employee <a href="#">[ View ]</a>	Review <a href="#">[ View ]</a>
Employment Website <a href="#">[ View ]</a>	Sessions <a href="#">[ View ]</a>
Flight <a href="#">[ View ]</a>	Student <a href="#">[ View ]</a>
Item <a href="#">[ View ]</a>	StudentSession <a href="#">[ View ]</a>
Jewel Customer <a href="#">[ View ]</a>	Technician <a href="#">[ View ]</a>
Job Application <a href="#">[ View ]</a>	Trainer <a href="#">[ View ]</a>
Knowledge <a href="#">[ View ]</a>	WorkOrder <a href="#">[ View ]</a>

7. Click on Edit, enable the check box for the status field.

WorkOrder Field-Level Security for profile Technician Help for

Field Name	Field Type	Read Access	Edit Access
Created By	Lookup	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Date	Date	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Description	Long Text Area	<input type="checkbox"/>	<input type="checkbox"/>
Email	Email	<input type="checkbox"/>	<input type="checkbox"/>
Last Modified By	Lookup	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Location	Picklist	<input type="checkbox"/>	<input type="checkbox"/>
Owner	Lookup	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Priority	Picklist	<input type="checkbox"/>	<input type="checkbox"/>
Service Type	Picklist	<input type="checkbox"/>	<input type="checkbox"/>
Status	Picklist	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
WorkOrder ID	Text	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

[Save](#) [Cancel](#)

8. Click on Save.

## Milestone-7: Users

Users are defined as the employees of your organization

### Create User

1. Go to setup --> type users in quick find box --> select users --> click New user.
2. Fill in the fields
1. First Name : Elina
2. Last Name : Gilbert
3. Alias : Give a Alias Name
4. Email id : Give your Personal Email id
5. Username : Username should be in this form: text@text.text
6. Nick Name : Give a Nickname
7. Role :
8. User license : Salesforce Platform
9. Profiles : Technician

New User

Help for this Page ?

User Edit

Save Save & New Cancel

General information

First Name: Elina  
Last Name: Gilbert  
Alias: elina  
Email: nadeem@thesmartbridge.co  
Username: elina@smart.com  
Nickname: elina  
Title:  
Company:  
Department:  
Division:

Role: <None Specified>  
User License: Salesforce Platform  
Profile: Technician  
Active: ✓

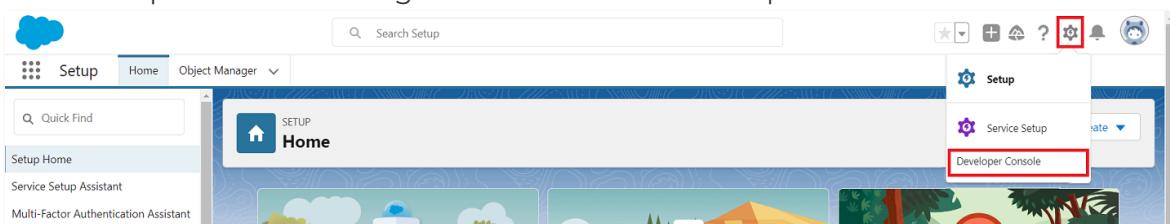
Marketing User  
Offline User  
Knowledge User  
Flow User  
Service Cloud User  
Site.com Contributor User  
Site.com Publisher User  
WDC User

3. Save.

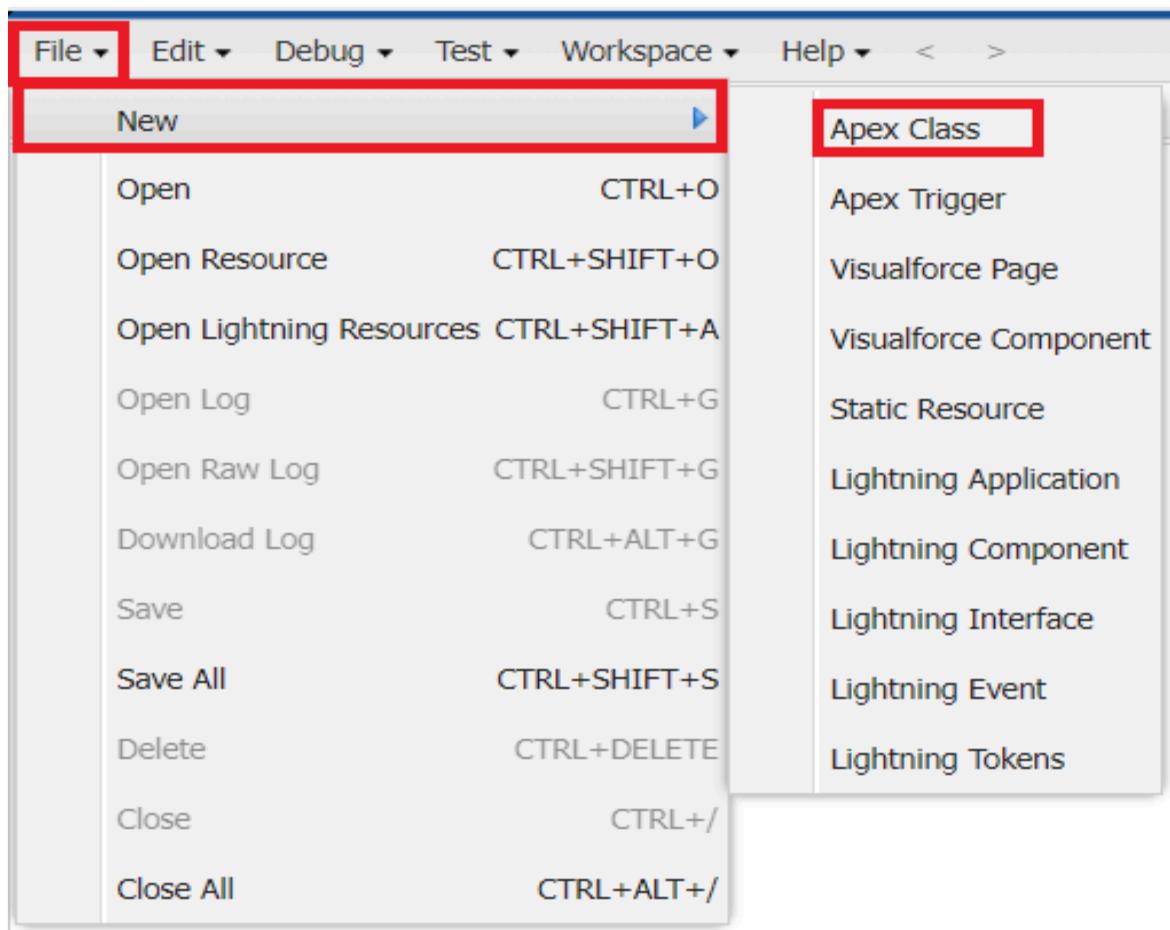
### Milestone-8: Apex Trigger

#### Create an Apex Class

1. Go to Setup --> Click on the gear icon --> Select Developer Console.



2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3. To create a new Apex Class follow the below steps:  
Click on the file --> New --> Apex Class.



4. Give the Apex Class name as "WorkOrderClass".



5. Click ok.
6. Now write the code logic here

```

1 public class WorkOrderClass {
2     public static void workOrder(List<WorkOrder__c> newListWorkOrder){
3         Map<Integer, List<String>> maptotech = new Map<Integer, List<String>>();
4         Integer num = 0;
5         List<WorkOrder__c> properWo = new List<WorkOrder__c>();
6         List<Assignment__c> lstAssignment = new List<Assignment__c>();
7         List<Technician__c> technicianAssignment = new List<Technician__c>();
8         for(WorkOrder__c iter : newListWorkOrder){
9             List<String> lststring = new List<String>();
10            If(iter.Service_Type__c != null && iter.Location__c != null ){
11                num = num+1;
12                properWo.add(iter);
13                lststring.add(iter.Service_Type__c);
14                lststring.add(iter.Location__c);
15
16                maptotech.put(num,lststring);
17            }
18        }
19        Map<integer,Id> techId = new Map<integer,Id>();
20        Map<Id,Technician__c> allTechnician = new Map<Id,Technician__c>([SELECT Id, Name, Phone__c, Location__c, Skills__c, Availability__c, Name__c, Email__c FROM Technician__c]);
21        Integer num2 = 0;
22        For(Technician__c T : allTechnician.values()){
23            num2 = num2+1;
24            If(maptotech.get(num2) != null){
25                List<String> valofmap = maptotech.get(num2);
26                system.debug('error 1 ----> the maptotech is empty ---> ' + maptotech.get(num2));
27                If(valofmap.contains(t.skills__c) && ValofMap.contains(t.Location__c) && t.Availability__c == 'Available'){
28                    techId.put(num2,t.Id);
29                }
30            }
31        }
32    }
33    integer num3 = 0;
34    For(WorkOrder__c iter : properWo){

```

### Source Code:

```

public class WorkOrderClass {
    public static void workOrder(List<WorkOrder__c> newListWorkOrder){
        Map<Integer, List<String>> maptotech = new Map<Integer, List<String>>();
        Integer num = 0;
        List<WorkOrder__c> properWo = new List<WorkOrder__c>();
        List<Assignment__c> lstAssignment = new List<Assignment__c>();
        List<Technician__c> technicianAssignment = new List<Technician__c>();
        for(WorkOrder__c iter : newListWorkOrder){
            List<String> lststring = new List<String>();
            If(iter.Service_Type__c != null && iter.Location__c != null ){
                num = num+1;
                properWo.add(iter);
                lststring.add(iter.Service_Type__c);
                lststring.add(iter.Location__c);

                maptotech.put(num,lststring);
            }
        }
        Map<integer,Id> techId = new Map<integer,Id>();
        Map<Id,Technician__c> allTechnician = new Map<Id,Technician__c>([SELECT
Id, Name, Phone__c, Location__c, Skills__c, Availability__c, Name__c, Email__c
FROM Technician__c]);
        Integer num2 = 0;
        For(Technician__c T : allTechnician.values()){
            num2 = num2+1;
            If(maptotech.get(num2) != null){
                List<String> valofmap = maptotech.get(num2);

```

```

        system.debug('error 1 ----> the maptotech is empty ---> ' +
maptotech.get(num2));
        if(valofMap.contains(t.Skills__c) && ValofMap.contains(t.Location__c) &&
t.Availability__c == 'Available'){
            techid.put(num2,t.Id);
        }
    }

}
integer num3 = 0;
For(WorkOrder__c W : properWo){
    num3 = num3 + 1;
    Assignment__c A = new Assignment__c();
    A.WorkOrder_ID__c = W.Id;
    A.Technician_ID__c = techid.get(num3);
    lstAssignment.add(A);
}
If(!lstAssignment.IsEmpty()){
    insert lstAssignment;
}
}
}
}

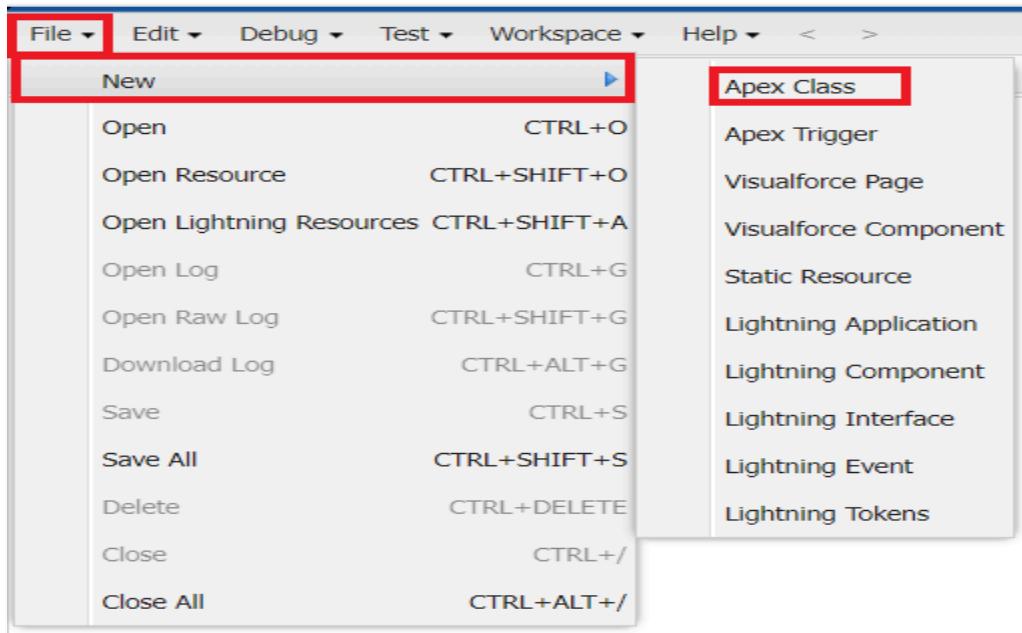
```

**Save the code.(click on file --> Save)**

### **Create an Apex Trigger**

1. To create a new Apex Class follow the below steps:

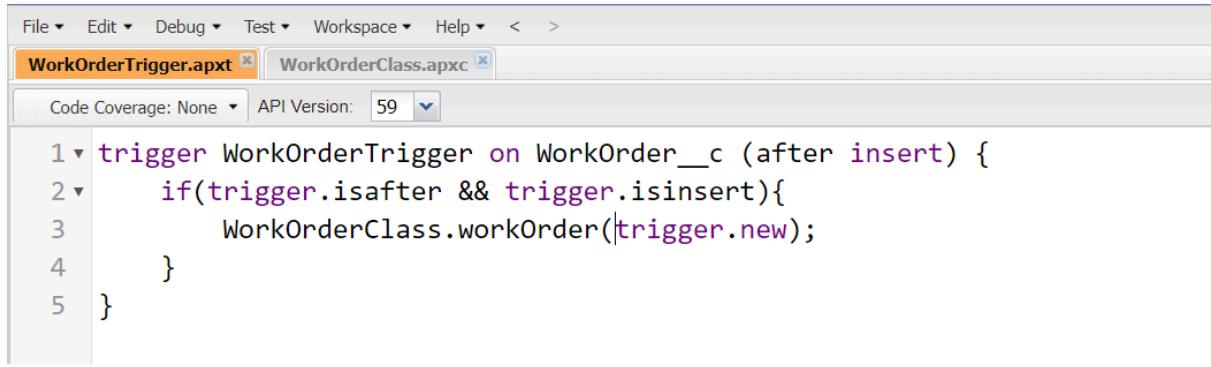
Click on the file --> New --> Apex Class.



2. Give the Apex Trigger name as "WorkOrderTrigger", and select "WorkOrder\_\_c" from the dropdown for sObject.

The screenshot shows a modal dialog box titled 'New Apex Trigger'. The dialog contains two input fields: 'Name:' with an empty text box and 'sObject:' with a dropdown menu showing a single option. At the bottom right of the dialog is a 'Submit' button, which is also highlighted with a red box.

3. Click Submit.
4. Now write the code logic here



The screenshot shows the Salesforce Developer Console interface. The top navigation bar includes File, Edit, Debug, Test, Workspace, Help, and tabs for WorkOrderTrigger.apxt and WorkOrderClass.apxc. Below the tabs, there are dropdowns for Code Coverage (None) and API Version (59). The main area displays the Apex trigger code:

```
1 trigger WorkOrderTrigger on WorkOrder__c (after insert) {  
2     if(trigger.isafter && trigger.isinsert){  
3         WorkOrderClass.workOrder(trigger.new);  
4     }  
5 }
```

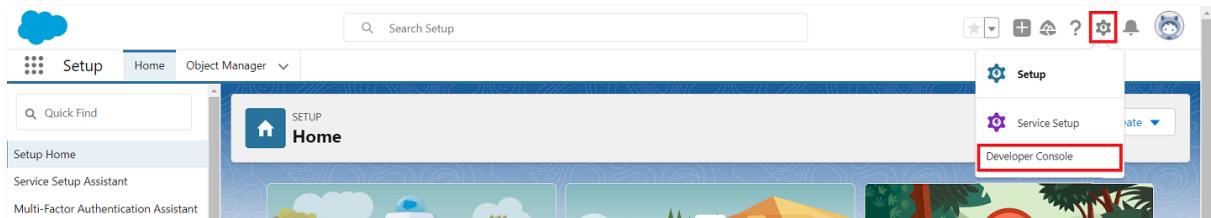
### Source Code:

```
trigger WorkOrderTrigger on WorkOrder__c (after insert) {  
    if(trigger.isafter && trigger.isinsert){  
        WorkOrderClass.workOrder(trigger.new);  
    }  
}
```

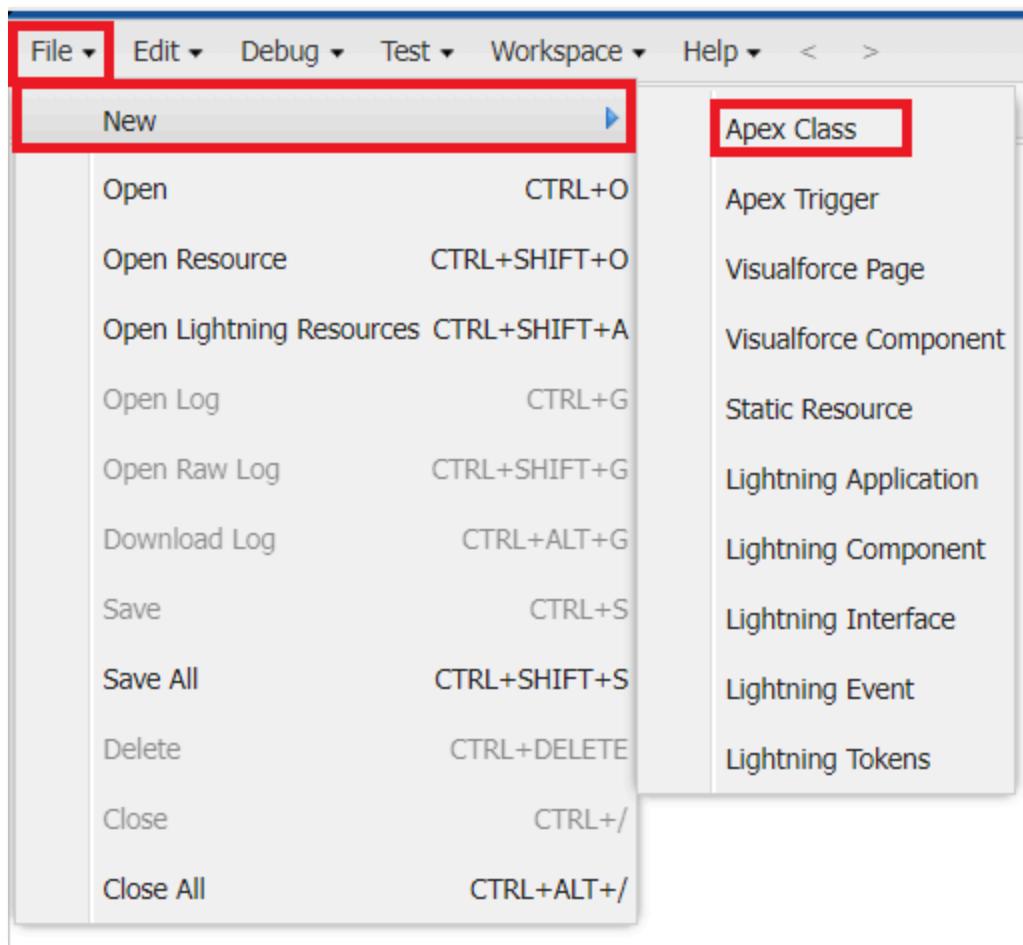
5. Save the code.(click on file --> Save)

### Create an Apex Class

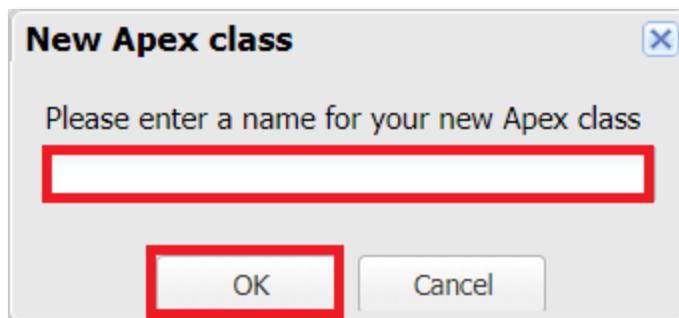
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3. To create a new Apex Class follow the below steps:  
Click on the file --> New --> Apex Class.



4. Give the Apex Class name as "AssigningEmail".



5. Click ok.
6. Now write the code logic here

```

1  public class AssigningEmail {
2    public static void sendEmailmsg(List<Assignment__c> assRec){
3      List<messaging.SingleEmailMessage> myVar = new List<messaging.SingleEmailMessage>();
4      Map<id,Technician__c> technicians = new Map<id,Technician__c>([SELECT Id, Phone__c, Location__c, Skills__c, Name__c, Email__c, Availability__c, Name FROM Technician__c]);
5      try{
6        for(Assignment__c con : assRec){
7          if(con.Technician_ID__c != null){
8            messaging.SingleEmailMessage mail = new messaging.SingleEmailMessage();
9            List<String> sendto = new List<String>();
10           sendto.add(technicians.Get(con.Technician_ID__c).Email__c);
11           mail.setToAddresses(sendto);
12           string subject = ' email sub';
13           mail.setSubject(subject);
14           string body = 'email body ';
15           mail.setHTMLBody(body);
16           myVar.add(mail);
17         }
18       }
19       Messaging.sendEmail(myVar);
20     }
21   catch(exception e){
22     system.debug('Error ----- ' + e.getMessage());
23   }
24 }
25
26 }

```

### Source Code:

```

public class AssigningEmail {
  public static void sendEmailmsg(List<Assignment__c> assRec){
    List<messaging.SingleEmailMessage> myVar = new
List<messaging.SingleEmailMessage>();
    Map<id,Technician__c> technicians = new Map<id,Technician__c>([SELECT Id,
Phone__c, Location__c, Skills__c, Name__c, Email__c, Availability__c, Name FROM
Technician__c]);
    try{
      for(Assignment__c con : assRec){
        if(con.Technician_ID__c != null){
          messaging.SingleEmailMessage mail = new
messaging.SingleEmailMessage();
          List<String> sendTo = new List<String>();
          sendTo.add(technicians.Get(con.Technician_ID__c).Email__c);
          mail.setToAddresses(sendTo);
          string subject = 'WorkOrder Assignment ';
          mail.setSubject(subject);
          string body = 'The following WorkOrder has been assigned to you ';
          mail.setHTMLbody(body);
          myVar.add(mail);
        }
      }
      Messaging.sendEmail(myVar);
    }
  }

```

```

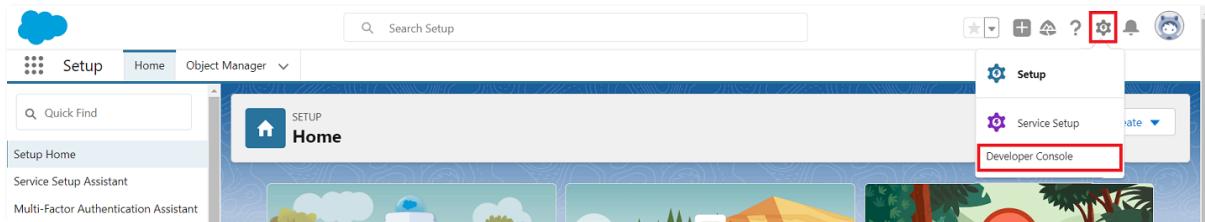
        catch(exception e){
            system.debug('Error -----> ' + e.getMessage());
        }
    }
}

```

- Save the code.(click on file --> Save)

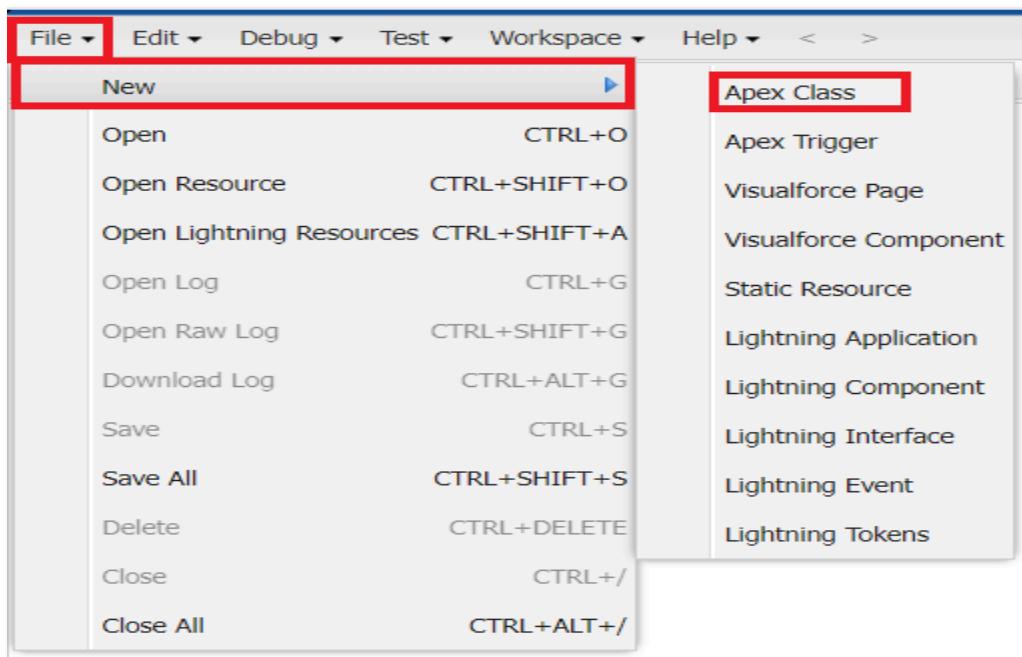
## Create an Apex Class

- Go to Setup --> Click on the gear icon --> Select Developer Console.

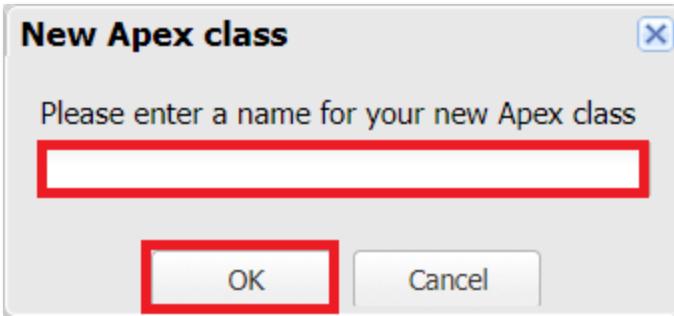


- Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
- To create a new Apex Class follow the below steps:

Click on the file --> New --> Apex Class.



- Give the Apex Class name as "AssigningEmail".



5. Click ok.
6. Now write the code logic here

```

1 * public class AssigningEmail {
2     public static void sendEmailmsg(List<Assignment__c> assRec){
3         List<messaging.SingleEmailMessage> myVar = new List<messaging.SingleEmailMessage>();
4         Map<id,Technician__c> technicians = new Map<id,Technician__c>([SELECT Id, Phone__c, Location__c, Skills__c, Name__c, Email__c, Availability__c, Name FROM Technician__c]);
5         try{
6             for(Assignment__c con : assRec){
7                 if(con.Technician_ID__c != null){
8                     messaging.SingleEmailMessage mail = new messaging.SingleEmailMessage();
9                     List<String> sendto = new List<String>();
10                    sendto.add(technicians.Get(con.Technician_ID__c).Email__c);
11                    mail.setToAddresses(sendto);
12                    string subject = ' email sub';
13                    mail.setSubject(subject);
14                    string body = 'email body ';
15                    mail.setHTMLBody(body);
16                    myVar.add(mail);
17                }
18            }
19            Messaging.sendEmail(myVar);
20        }
21        catch(exception e){
22            system.debug('Error ----> ' + e.getMessage());
23        }
24    }
25 }
26

```

#### Source Code:

```

public class AssigningEmail {

    public static void sendEmailmsg(List<Assignment__c> assRec){
        List<messaging.SingleEmailMessage> myVar = new
List<messaging.SingleEmailMessage>();
        Map<id,Technician__c> technicians = new Map<id,Technician__c>([SELECT Id,
Phone__c, Location__c, Skills__c, Name__c, Email__c, Availability__c, Name FROM
Technician__c]);
        try{
            for(Assignment__c con : assRec){
                if(con.Technician_ID__c != null){
                    messaging.SingleEmailMessage mail = new
messaging.SingleEmailMessage();
                    List<String> sendTo = new List<String>();
                    sendTo.add(technicians.Get(con.Technician_ID__c).Email__c);
                    mail.setToAddresses(sendTo);

```

```

        string subject = 'WorkOrder Assignment ';
        mail.setSubject(subject);
        string body = 'The following WorkOrder has been assigned to you ';
        mail.setHTMLbody(body);
        myVar.add(mail);
    }
}

Messaging.sendEmail(myvar);
}
catch(exception e){
    system.debug('Error -----> ' + e.getMessage());
}
}
}

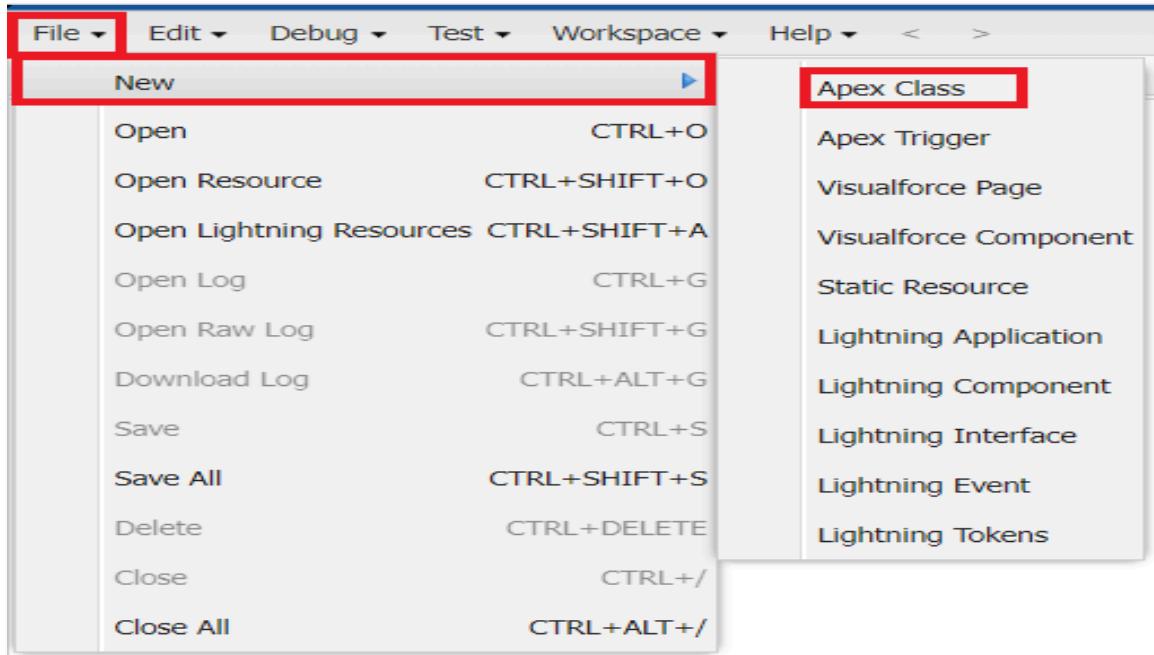
```

7. Save the code.(click on file --> Save)

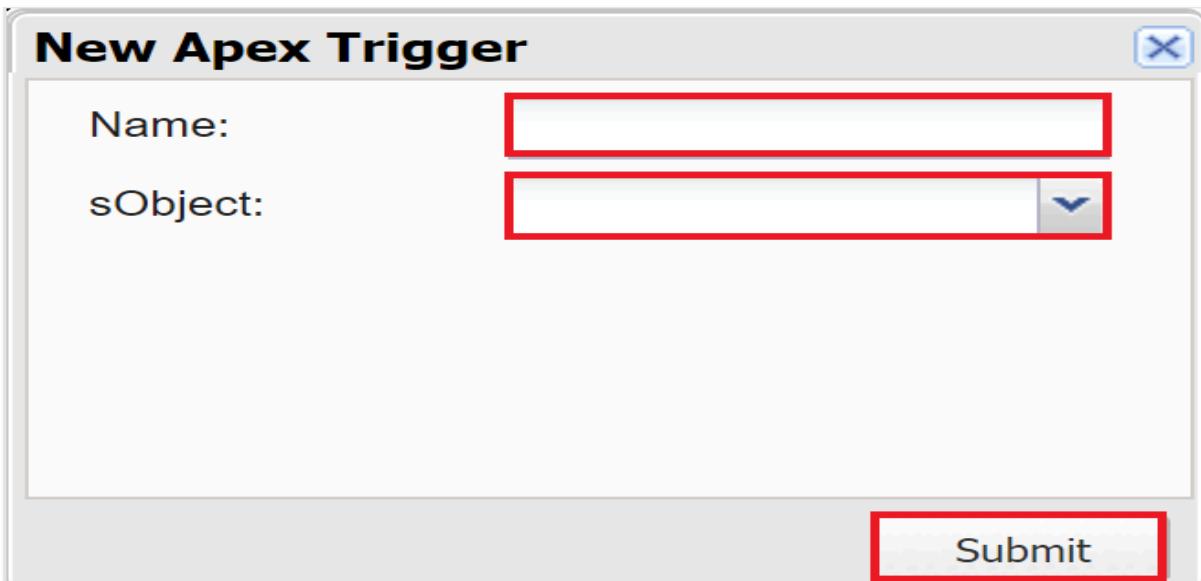
## Create an Apex Trigger

To create a new Apex Class follow the below steps:

1. Click on the file --> New --> Apex Class.



2. Give the Apex Trigger name as "AssignmentTrigger", and select "Assignment\_\_c" from the dropdown for sObject.



3. Click Submit.
4. Now write the code logic here

```
RecordDeletions.apxc ScheduleClass.apxc AssignmentTrigger.apxt
Code Coverage: None API Version: 59
1 trigger AssignmentTrigger on Assignment__c (after insert) {
2     if(Trigger.IsAfter && Trigger.IsInsert){
3         AssigningEmail.sendEmailmsg(Trigger.New);
4     }
5 }
```

A screenshot of the Salesforce developer console. The tab bar at the top shows 'AssignmentTrigger.apxt' is active. Below the tabs are dropdowns for 'Code Coverage: None' and 'API Version: 59'. The main area contains the Apex trigger code shown in the previous image.

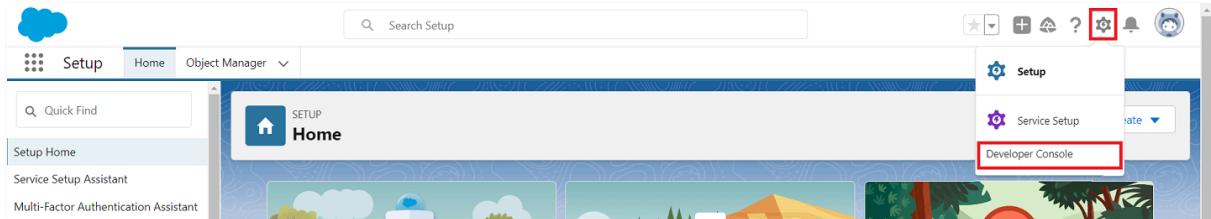
#### Source Code:

```
trigger AssignmentTrigger on Assignment__c (after insert) {
    if(Trigger.IsAfter && Trigger.IsInsert){
        AssigningEmail.sendEmailmsg(Trigger.New);
    }
}
```

Save the code.(click on file --> Save)

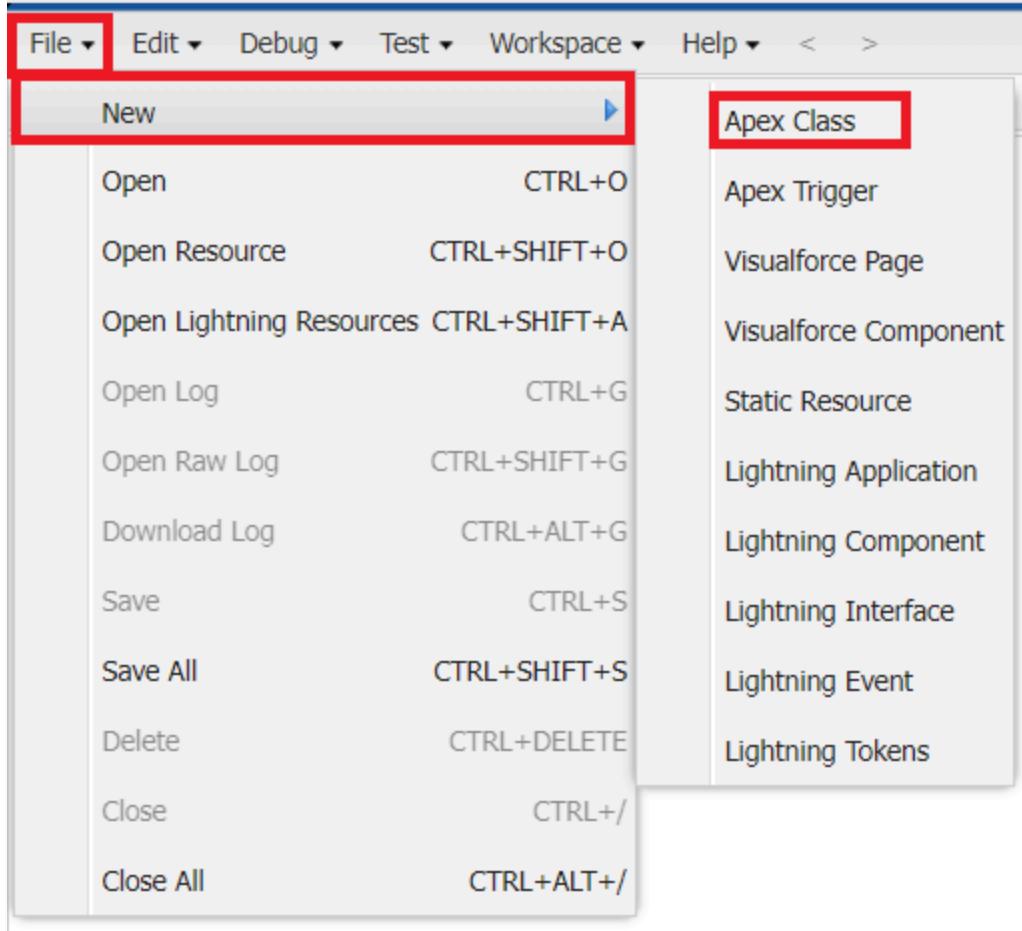
#### Create an Apex Class

1. Go to Setup --> Click on the gear icon --> Select Developer Console.

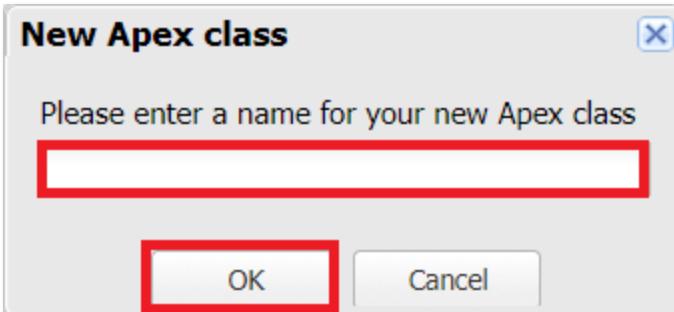


2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3. To create a new Apex Class follow the below steps:

Click on the file --> New --> Apex Class.



4. Give the Apex Class name as "CompletionMail".



5. Click ok.
6. Now write the code logic here

Screenshot of the Salesforce code editor showing the 'CompletionMail.apxc' file. The code implements a class named 'CompletionMail' with a static method 'sendEmailMsg' that sends an email to resolved work orders.

```

RecordDeletions.apxc ScheduleClass.apxc CompletionMail.apxc
Code Coverage: None API Version: 59
1 public class CompletionMail {
2     public static void sendEmailMsg(List<WorkOrder__c> workOrderList){
3         List<messaging.SingleEmailMessage> myVar = new List<messaging.SingleEmailMessage>();
4         for(WorkOrder__c con : workOrderList){
5             if(con.Status__c == 'Resolved'){
6                 messaging.SingleEmailMessage mail = new messaging.SingleEmailMessage();
7                 List<String> sendTo = new List<String>();
8                 sendTo.add(con.Email__c);
9                 mail.setToAddresses(sendTo);
10                string subject = 'Status Updated';
11                mail.setSubject(subject);
12                string body = 'email body ';
13                mail.setHTMLBody(body);
14                myVar.add(mail);
15            }
16        }
17        Messaging.sendEmail(myVar);
18    }
19 }

```

#### **Source Code:**

```

public class CompletionMail {
    public static void sendEmailMsg(List<WorkOrder__c> workOrderList){
        List<messaging.SingleEmailMessage> myVar = new
        List<messaging.SingleEmailMessage>();
        for(WorkOrder__c con : workOrderList){
            if(con.Status__c == 'Resolved'){
                messaging.SingleEmailMessage mail = new
                messaging.SingleEmailMessage();
                List<String> sendTo = new List<String>();
                sendTo.add(con.Email__c);
                mail.setToAddresses(sendTo);
                string subject = 'Status Updated';

```

```

        mail.setSubject(subject);
        string body = 'email body ';
        mail.setHTMLbody(body);
        myVar.add(mail);
    }
}
Messaging.sendEmail(myvar);
}
}

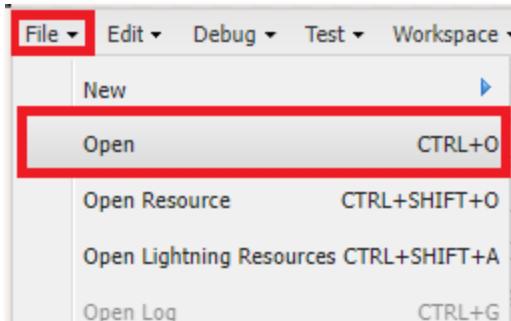
```

7.

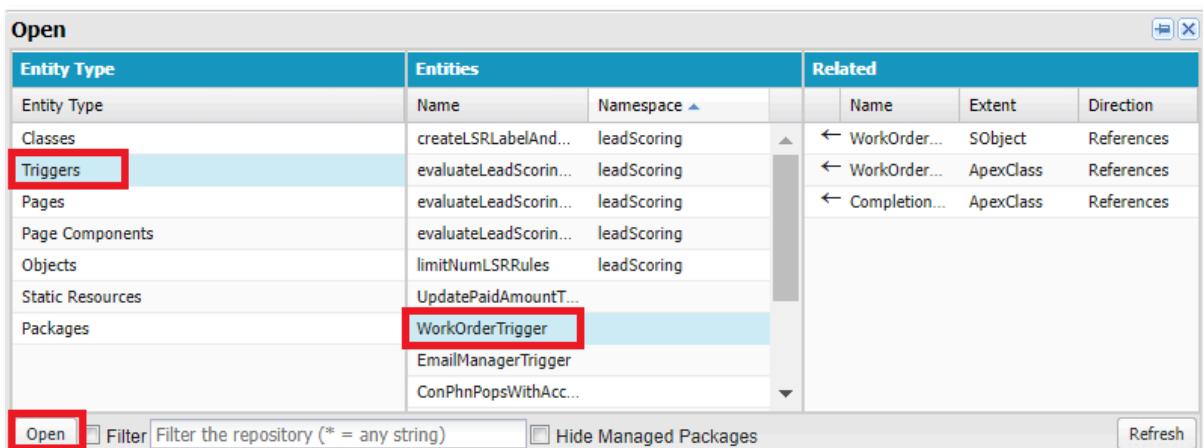
8. Save the code.(click on file --> Save)

## Create an Apex Trigger

1. Click on the file --> Open.



2. A pop up window opens click on Triggers, then select “WorkOrderTrigger” and click on “Open”



3. Now write the code logic here.

```

1 trigger WorkOrderTrigger on WorkOrder__c (after insert, after update) {
2     if(Trigger.IsAfter && Trigger.IsInsert){
3         WorkOrderClass.workOrder(trigger.new);
4     }
5     if(Trigger.IsAfter && Trigger.IsUpdate){
6         CompletionMail.sendEmailMsg(Trigger.New);
7     }
8 }

```

### Source Code:

trigger WorkOrderTrigger on WorkOrder\_\_c (after insert, after update) {

    if(Trigger.IsAfter && Trigger.IsInsert){

        WorkOrderClass.workOrder(trigger.new);

    }

    if(Trigger.IsAfter && Trigger.IsUpdate){

        CompletionMail.sendEmailMsg(Trigger.New);

    }

4. Save the code.(click on file --> Save)

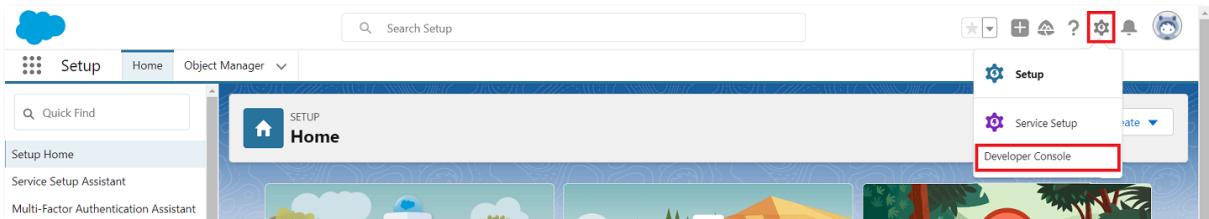
### Create an Asynchronous Apex Class

Create an Apex Class to Delete all the WorkOrder records which meets the following criteriaL

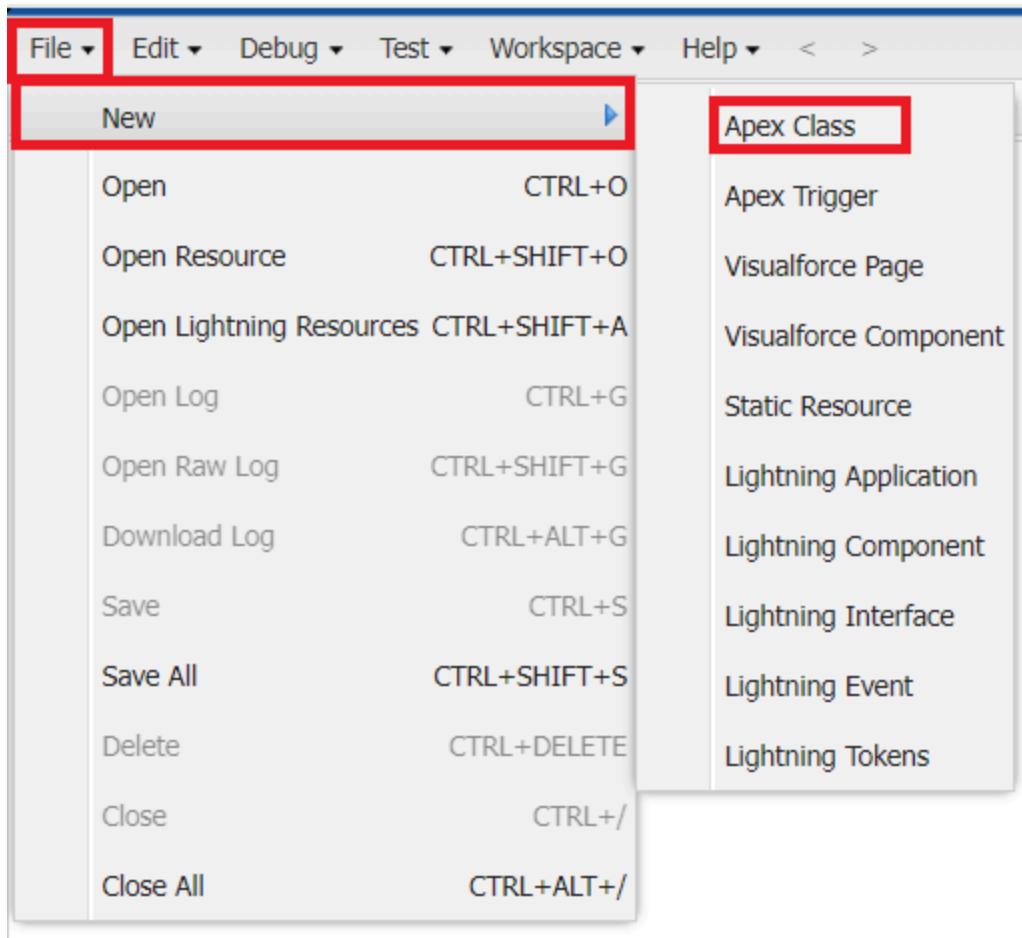
1. Completed date should be more than 30 days.
2. Status should be 'Resolved'.

### Create an Apex Class

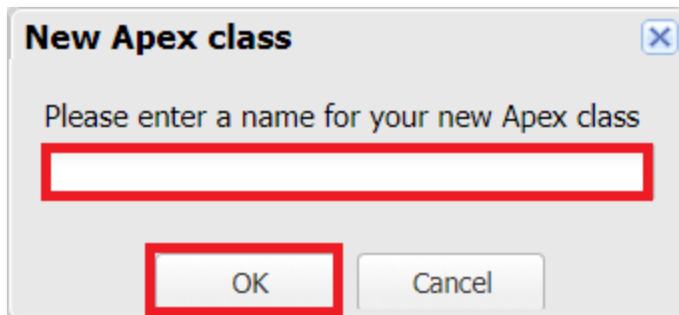
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3. To create a new Apex Class follow the below steps:  
Click on the file --> New --> Apex Class.



4. Give the Apex Class name as "RecordDeletion".



5. Click ok.
6. Now write the code logic here

```

1 * public class RecordDeletions Implements Database.Batchable<SObject>{
2     public Database.QueryLocator start(Database.BatchableContext bc) {
3         string query = 'SELECT Id, Name, WorkOrder_ID__c, Technician_ID__c, Assignment_Date__c, Completion_Date__c FROM Assignment__c WHERE Completion_Date__c = LAST_N_DAYS:30';
4         return database.GetQueryLocator(query);
5     }
6     public void execute(Database.BatchableContext bc, List<Assignment__c> query){
7         if(!Query.IsEmpty()){
8             Delete Query;
9         }
10    }
11    public void finish(Database.BatchableContext bc){
12    }
13 }
14

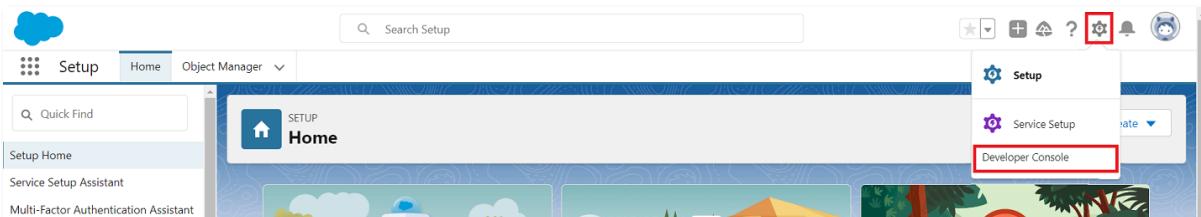
```

### Source Code:

7. public class RecordDeletions Implements Database.Batchable<SObject>{
8. public Database.QueryLocator start(Database.BatchableContext bc) {
9. string query = 'SELECT Id, Name, WorkOrder\_ID\_\_c, Technician\_ID\_\_c,  
Assignment\_Date\_\_c, Completion\_Date\_\_c FROM Assignment\_\_c WHERE  
Completion\_Date\_\_c = LAST\_N\_DAYS:30';
10. return database.GetQueryLocator(query);
11. }
12. public void execute(Database.BatchableContext bc, List<Assignment\_\_c>  
query){
13. if(!Query.IsEmpty()){
14. Delete Query;
15. }
16. }
17. public void finish(Database.BatchableContext bc){
18. }
19. }
- 20.
21. Save the code.(click on file --> Save)

### Create an Apex Schedule Class

1. Go to Setup --> Click on the gear icon --> Select Developer Console.

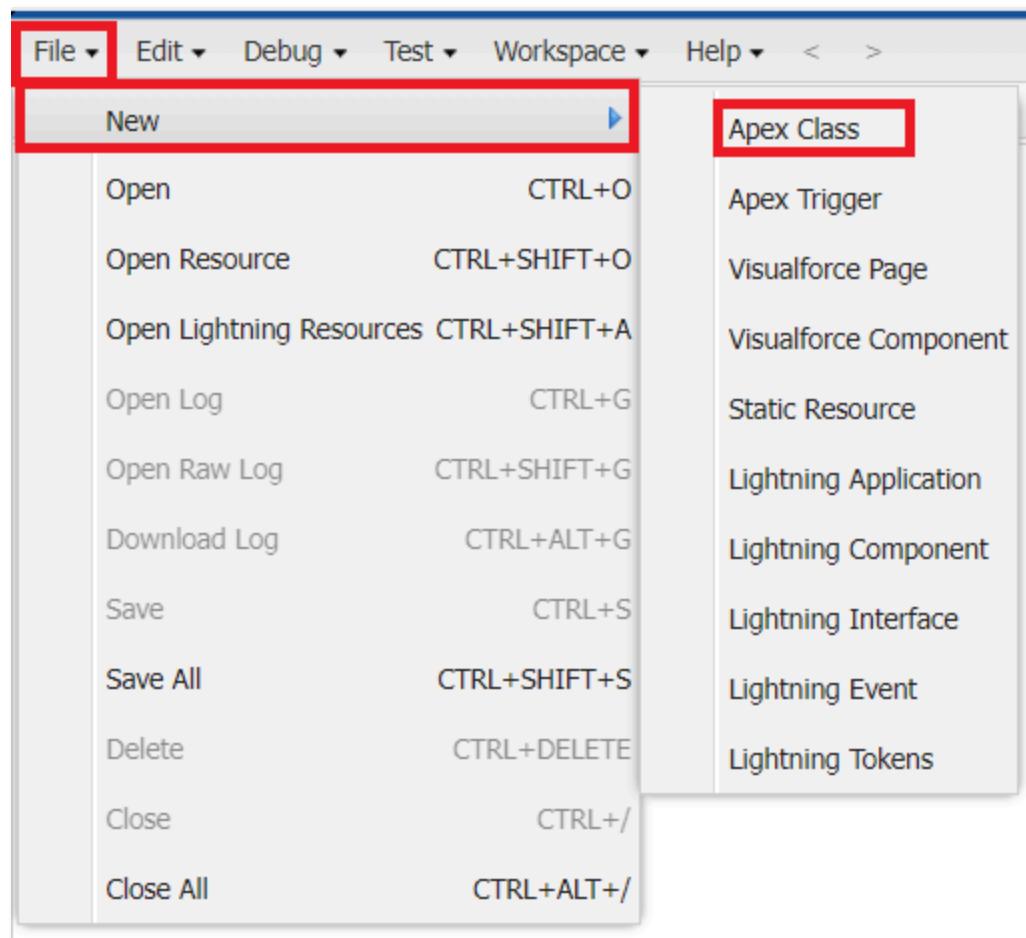


2. Then we can see the Developer console. Click on the developer console and you

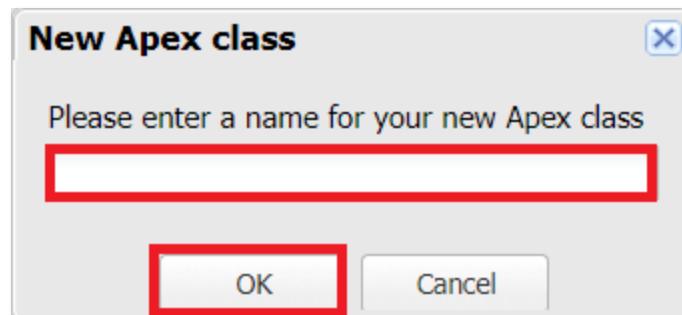
will navigate to a new console window.

3. To create a new Apex Class follow the below steps:

Click on the file --> New --> Apex Class.



4. Give the Apex Class name as "ScheduleClass".



5. Click ok.

6. Now write the code logic here

```

1 global class ScheduleClass implements Schedulable {
2     global void execute(SchedulableContext SC) {
3         RecordDeletions delrec = new RecordDeletions();
4         database.executeBatch(delrec, 200);
5     }
6 }

```

### Source Code:

```

global class ScheduleClass implements Schedulable {
    global void execute(SchedulableContext SC) {
        RecordDeletions delrec = new RecordDeletions();
        database.executeBatch(delrec, 200);
    }
}

```

- Save the code.(click on file ? Save)

## Create a Schedule Apex

Schedule the Apex class:

- From the Setup page search for “Apex Classes” in quick search.
- Click on “Schedule Apex” as shown below.

**Apex Classes**

Apex Code is an object oriented programming language that allows developers to develop on-demand business applications on the Lightning Platform.

**Percent of Apex Used: 0.19%**  
You are currently using 11,223 characters of Apex Code (excluding comments and @isTest annotated classes) in your organization, out of an allowed limit of 6,000,000 characters. Note that the amount in use includes both Apex Classes and Triggers defined in your organization.

Estimate your organization's code coverage [i](#)  
Compile all classes [i](#)  
View: All [Create New View](#)

Action	Name	Namespace Prefix	API Version	Status	Size Without Comments	Last Modified By	Has Trace Flags
Edit   Del   Security	Apex101		59.0	Active	181	demo.project	26/10/2023, 12:38 pm

- Click on Schedule Apex and enter the Job name.
  - Job Name : DeleteAssignmentSchedule
  - Apex Class : ScheduleClass (from clicking on lookup icon)

3. Frequency : Monthly
4. Preferred Start Time : Select any time

Schedule Apex

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

Job Name	<input type="text" value="DeleteAssignmentSchedule"/>	Save	Cancel
Apex Class	<input type="text" value="ScheduleClass"/>		
Schedule Apex Execution			
Frequency	<input type="radio"/> Weekly <input checked="" type="radio"/> Monthly	<input type="radio"/> On day <input type="text" value="1"/> of every month	<input type="radio"/> On <input type="text" value="the 1st"/> <input type="radio"/> Sunday of every month
Start	<input type="text" value="06/12/2023"/>	<input type="text" value="06/12/2023"/>	
End	<input type="text" value="06/01/2024"/>	<input type="text" value="06/12/2023"/>	
Preferred Start Time	<input type="text" value="4:00 pm"/>		
Exact start time will depend on job queue activity.			
<input type="button" value="Save"/> <input type="button" value="Cancel"/>			

4. Click Save.

## Milestone-9: Reports & Dashboards

Salesforce Reports and Dashboards are powerful tools that empower users to visualize and analyze data within the Salesforce platform. They play a crucial role in providing insights, monitoring performance, and making informed business decisions.

### Report

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

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

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

The screenshot shows the 'Create Report' interface. On the left, there's a 'Category' sidebar with various report types listed. A red box labeled '1' highlights the 'All' button. In the center, a search bar contains the text 'assignment' with a red box labeled '2'. Below the search bar is a list of 'Report Type Name' and 'Category' for each item. A red box labeled '3' highlights the 'Assignments with WorkOrder ID' entry. On the right, a 'Details' panel shows a report titled 'Assignments with WorkOrder ID' (Standard Report Type) with a red box labeled '4' over the title. Below it are sections for 'Created By You' and 'Created By Others', both showing 'No Reports Yet'. At the bottom, there's a section for 'Objects Used in Report Type' with icons for 'Owner' and 'WorkOrder'.

4. Customize your report

- Add fields from left pane as shown below
- Grouped by workorder ID

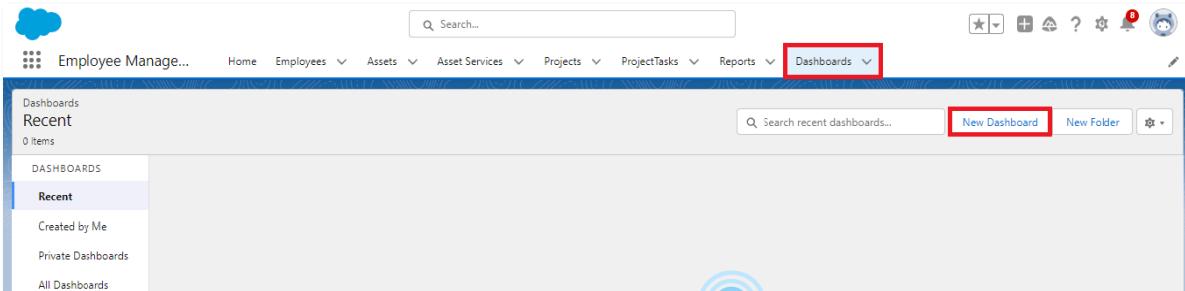
The screenshot shows the 'New Employees Report' configuration screen. On the left, there's a 'Fields' pane with sections for 'Outline', 'Groups', and 'Columns'. A red box highlights the 'Add group...' button under Groups and the 'Add column...' button under Columns. To the right, there's a preview area showing a table with two rows of employee data. The columns are: Employee, Employee Name, Employee ID, Reports to, Login Time, Logout Time, Mode of Work, and LinkedIn Profile. The first row is for 'Employee' and the second for 'Emp for Junction test'. A red box highlights the preview table. At the top right, there are buttons for 'Save & Run', 'Save', 'Close', and 'Run'.

5. Save or run it.

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

## Dashboard

1. Go to the app --> click on the Dashboards tabs.



2. Give a Name and click on Create.

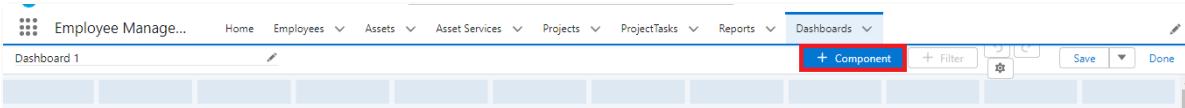
New Dashboard

\* Name  
Dashboard 1

Description

Folder  
Private Dashboards

3. Select add component.



4. Select a Report which we have created in the previous activities and click on select.

Select Report

Reports

Recent

Created by Me

Private Reports

Public Reports

All Reports

Folders

Created by Me

Shared with Me

Select Report

Employee's working on projects report  
Employee Project · 05-Jun-2023, 9:37 am · Private Reports

Assets assigned to Employees  
Employee Project · 05-Jun-2023, 9:36 am · Private Reports

- 5.

