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, availibility, 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: Lightining App Creation

Milestone-5: Fields & Relationships

Milestone-6: Profiles

Milestone-7: Users

Milestone-8: Apex Trigger

Milestone-9: Reports & Dashboards

Milestone 1- Creatione 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.

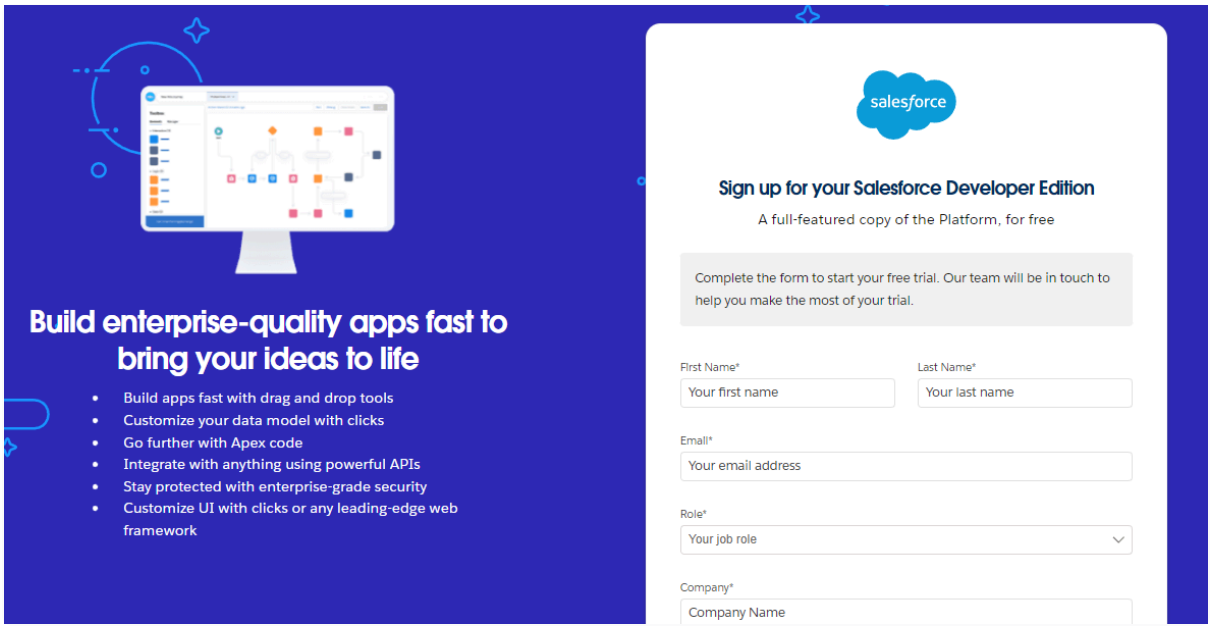
Salesforce services allow businesses to use cloud technology to better connect with partners, customers, and potential customers. Using the Salesforce CRM, companies can track customer activity, market to customers, and many more services.

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 :



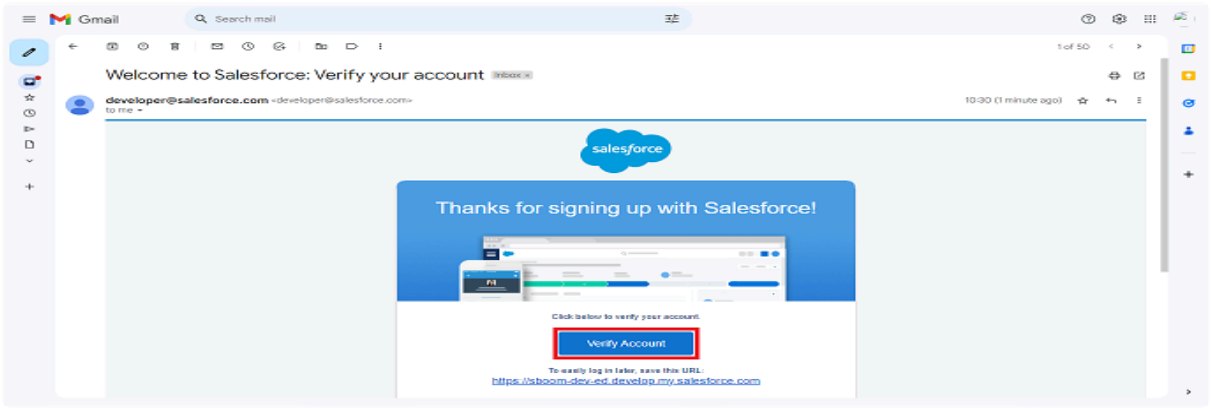
1. First name & Last name
2. Email
3. Role : Developer
4. Company : College Name
5. County : India
6. Postal Code : pin code
7. Username : should be a combination of your name and company

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

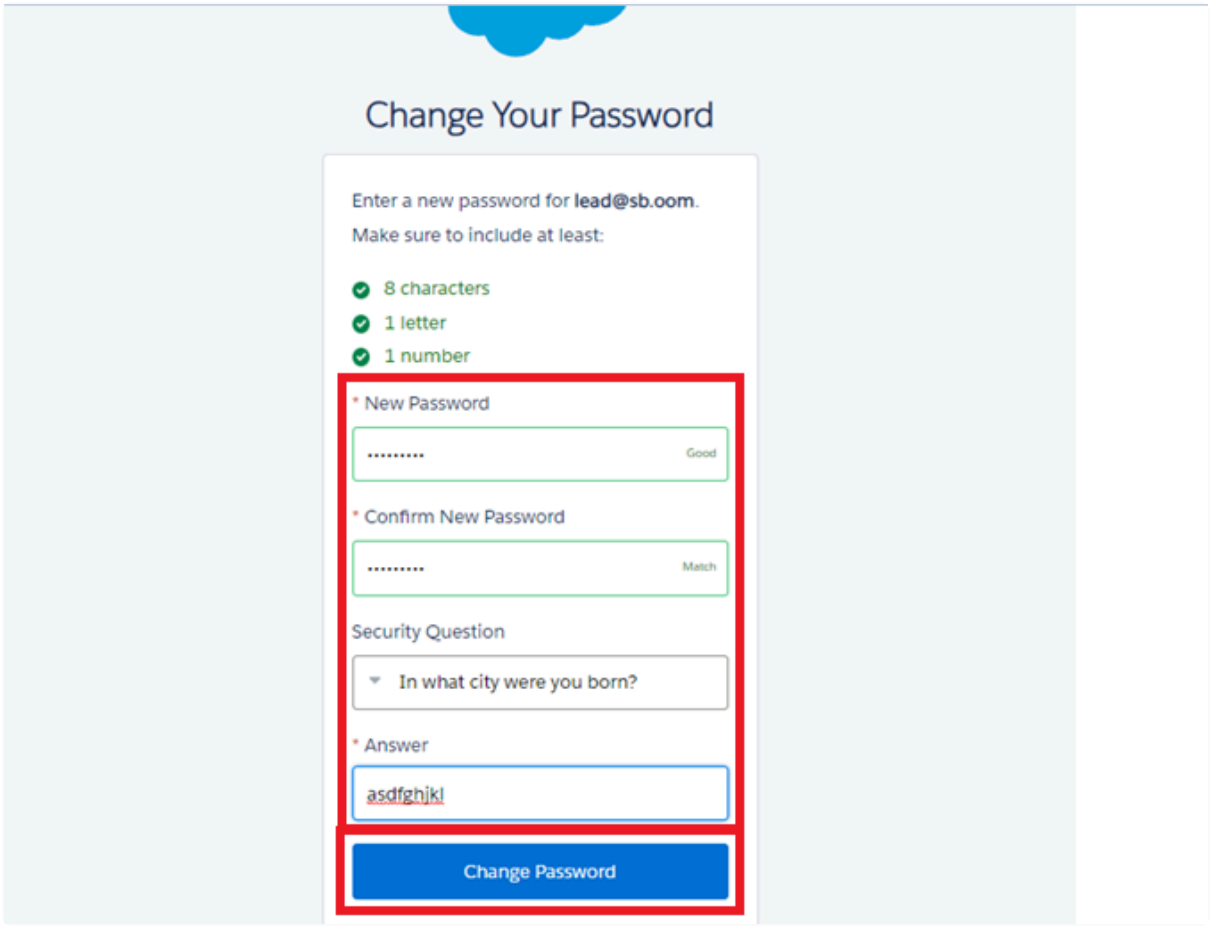
Click on sign me up after filling these.

Task 2: Account Activation

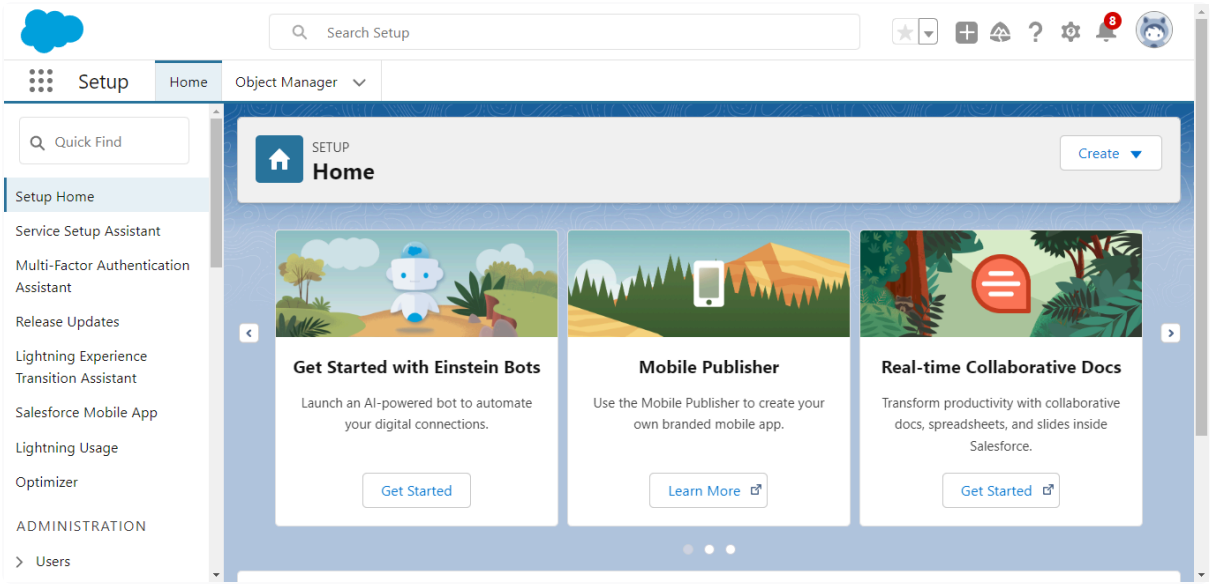
Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins.



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



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



Milestone 2- Object Creation

Objects are database tables that allow you to store data specific to an organization.
Salesforce objects are of two types:

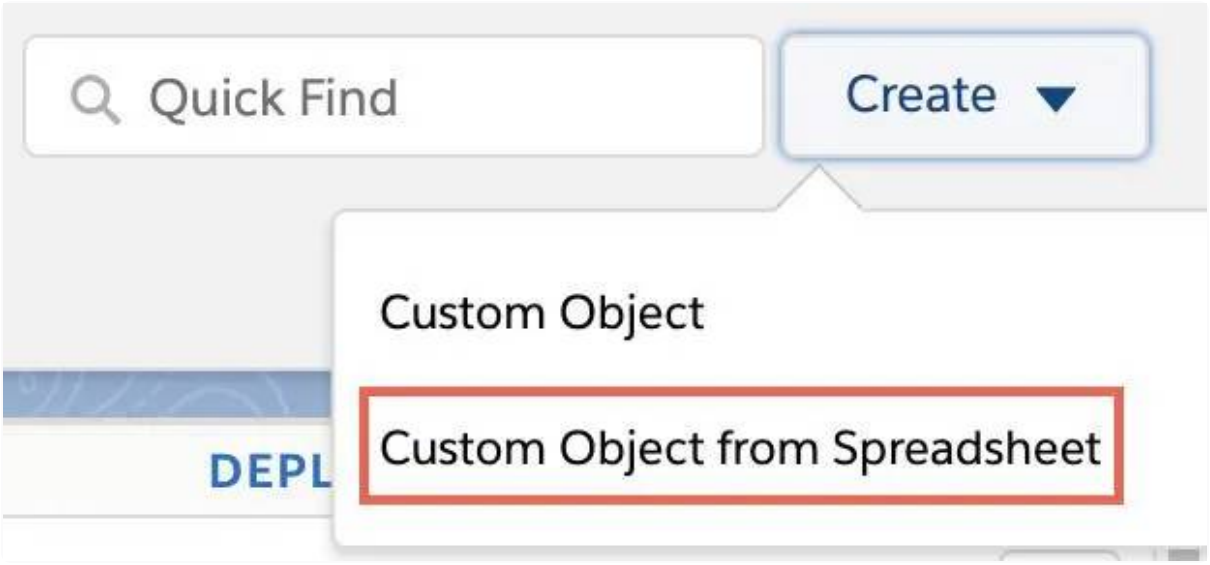
- **Standard Objects:** Standard objects are the kind of objects that are provided by salesforce.com such as users, contracts, reports, dashboards, etc.
- **Custom Objects:** Custom objects are created by users. They collect information that is unique and essential to their organization. They are the 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.

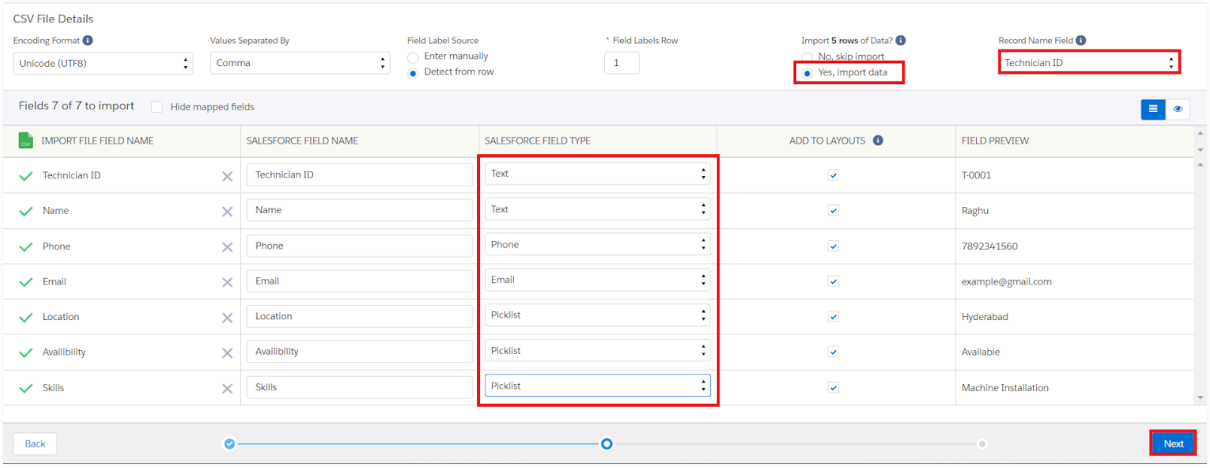
1. Log into your salesforce account, click on Gear icon, then select Setup.
2. Click the Object Manager tab.

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.

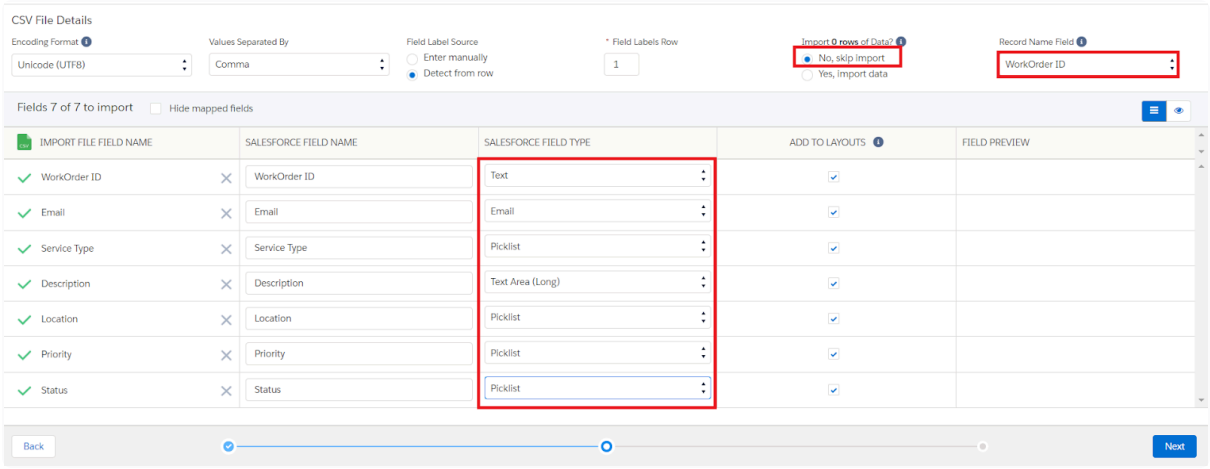


11. Click Next and enter the following settings.
12. Click Finish. The Technician object is successfully created and data imported, all within minutes.

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.



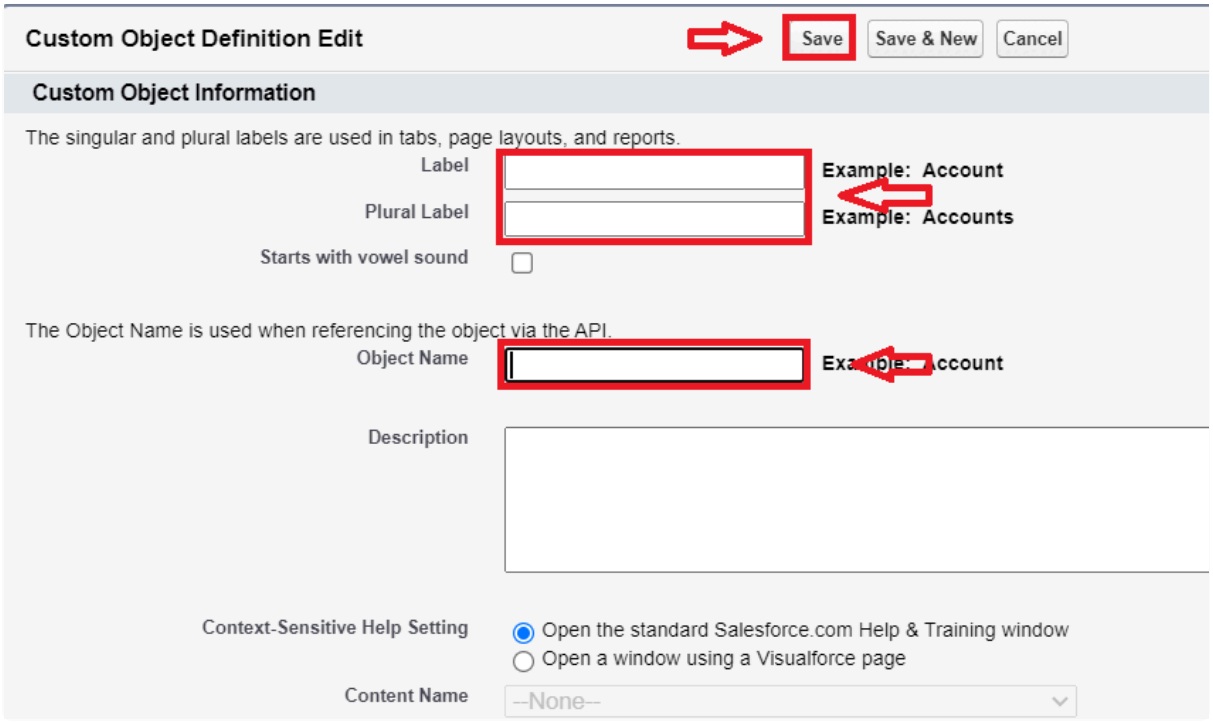
Task 3: Assignment Object

To create an object:

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

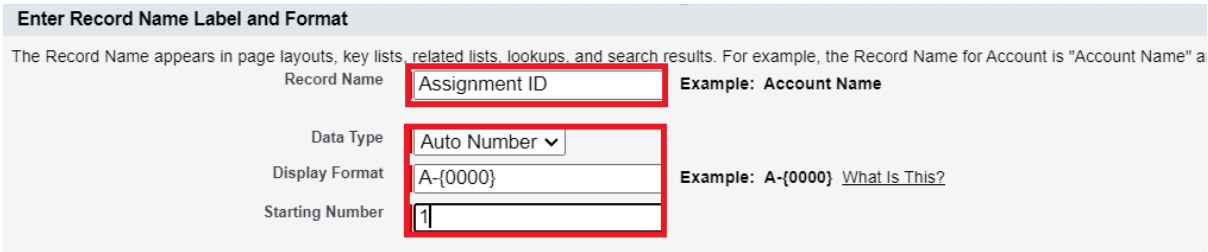
on Custom Object.

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



3. Enter Record Name Label and Format

- Record Name --> Assignment ID
- Data Type --> Auto Number
- Display Format --> A-{0000}
- Starting Number --> 1



4. Click on Allow reports,

Allow search --> Save.

Milestone 3- Tabs Creation

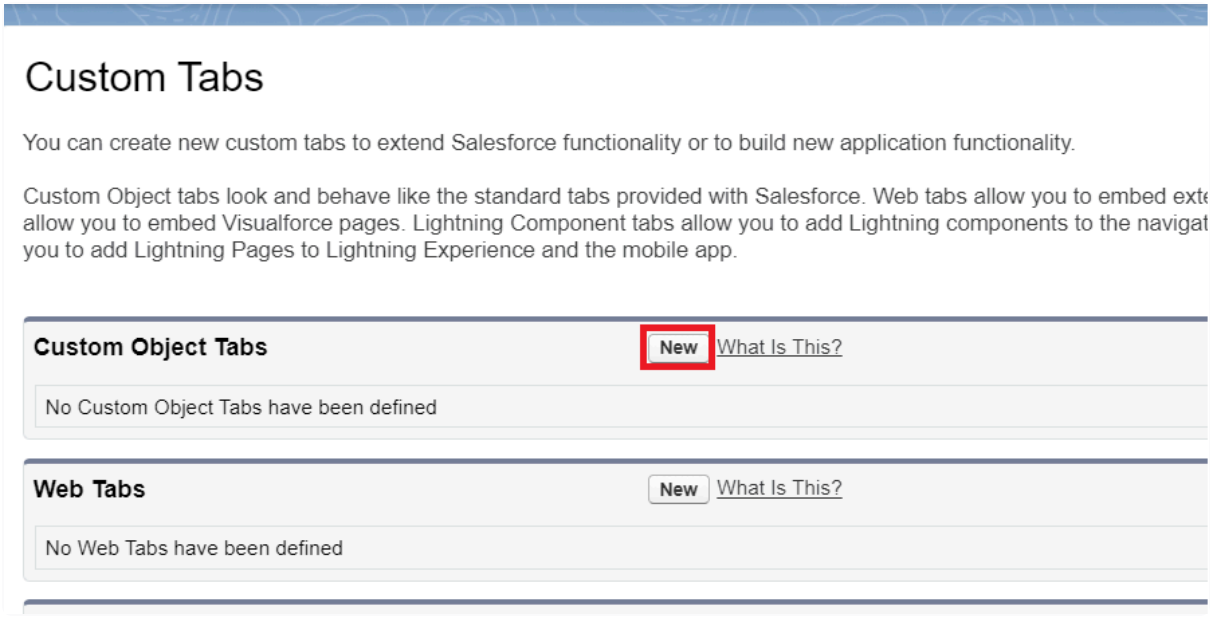
Tabs are the user interface elements that provide access to objects, functions, or data. They represent the front end of your Salesforce experience.

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

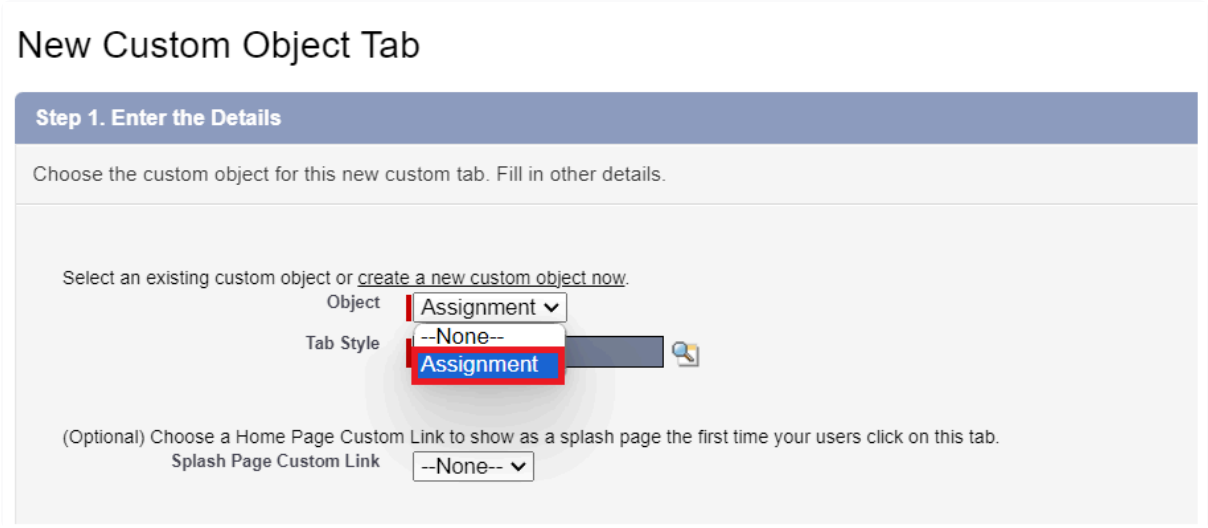
Task 1: Creating a Custom Tab

To create a Tab:(Assignment)

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



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.



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

Milestone 4- Lightining App Creation

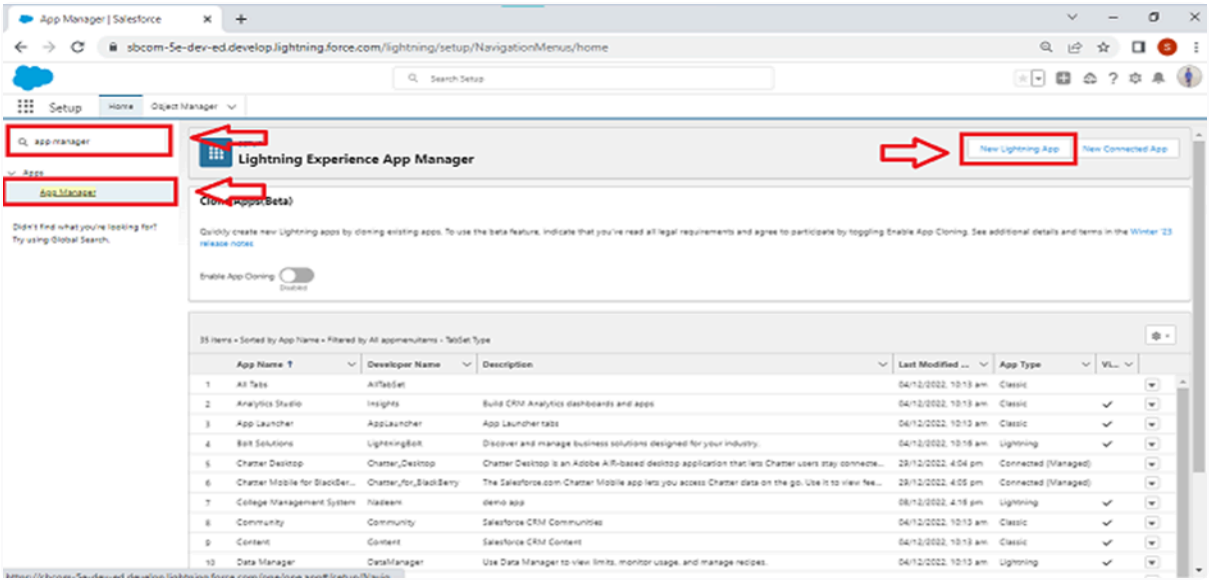
An app is a collection of items that work together to serve a particular function. In Lightning Experience, Lightning apps give your users access to sets of objects, tabs, and other items all in one convenient bundle in the navigation bar.

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.

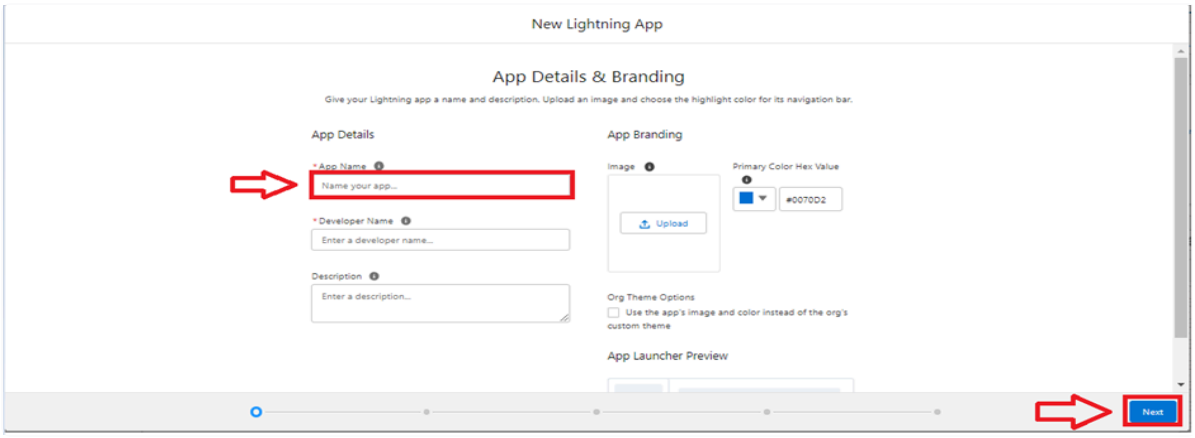
Task 1: Create a lightning app

To create a lightning app page:

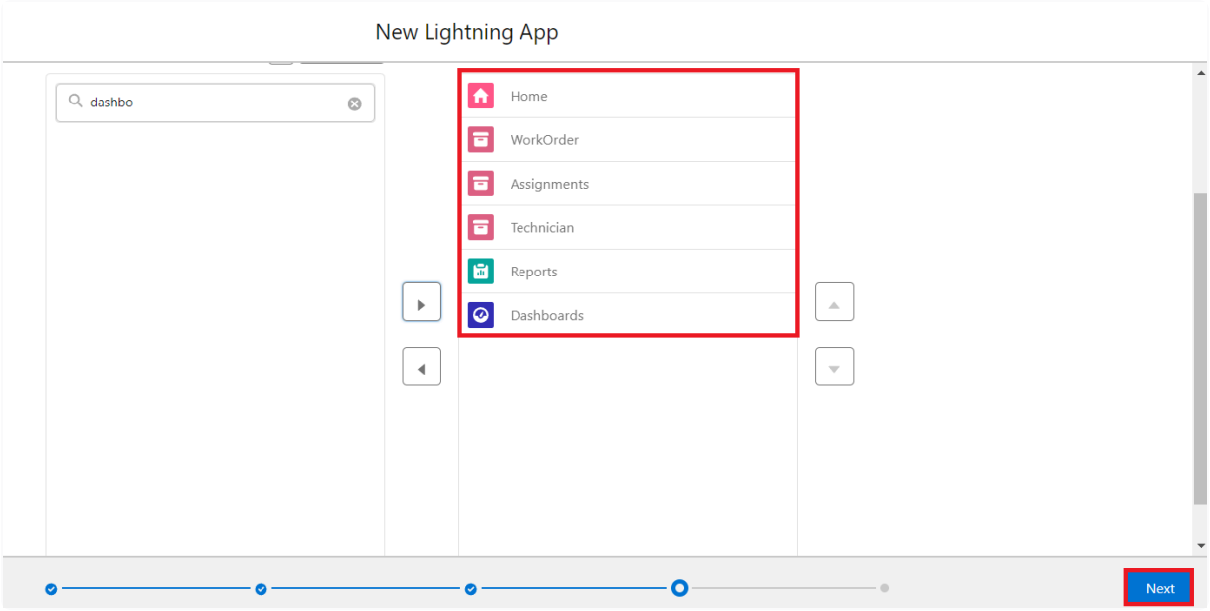
1. Go to setup page --> search “app manager” in quick find --> select “app manager” --> click on New lightning App.



2. 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
3. Then click Next --> (App option page) keep it as default --> Next --> (Utility Items) keep it as default --> Next.



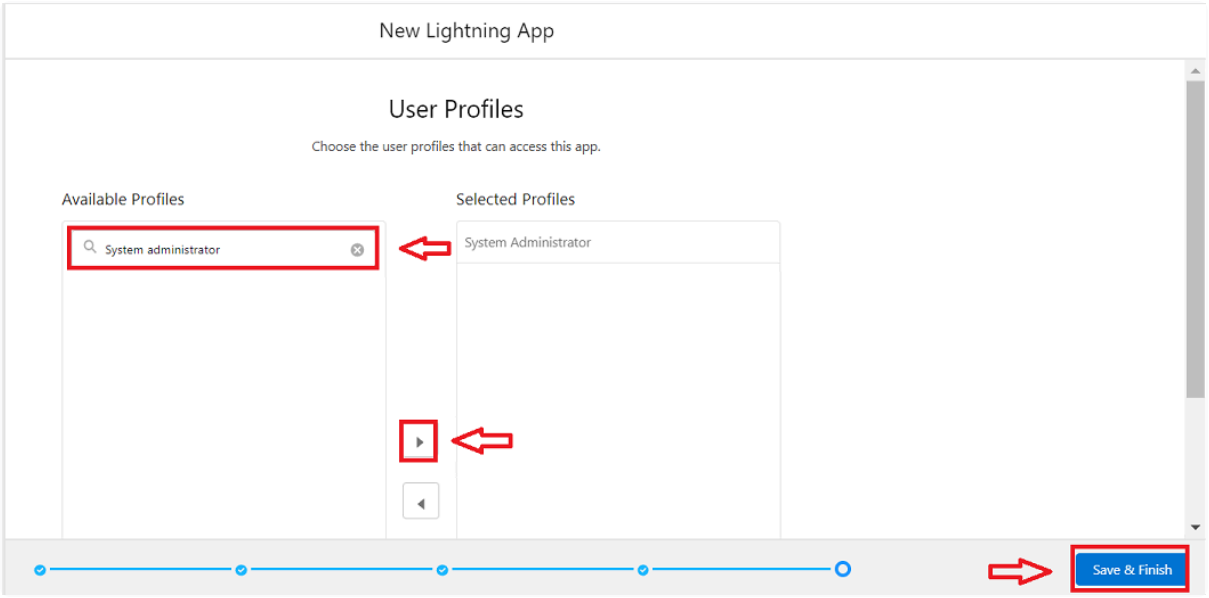
4. To Add Navigation Items:



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.

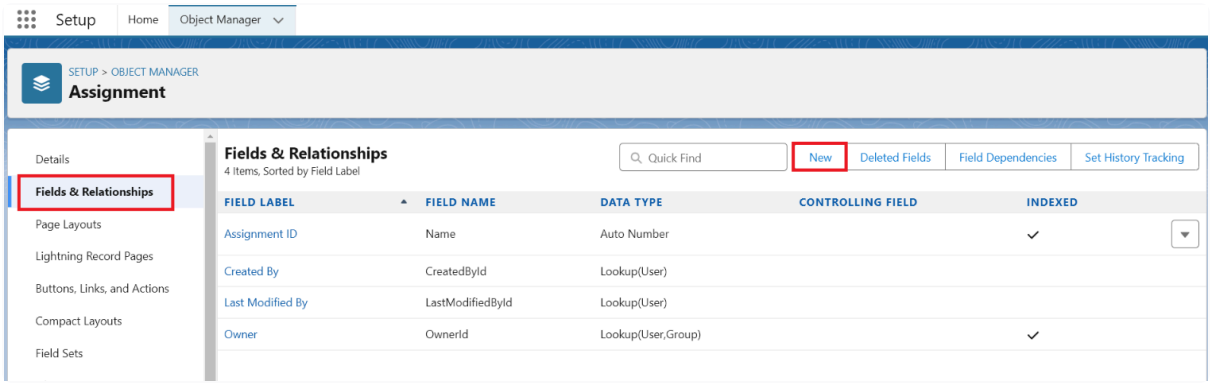
Milestone 5- Fields & Relationships

Task 1: Creating Lookup Field in Assignment Object

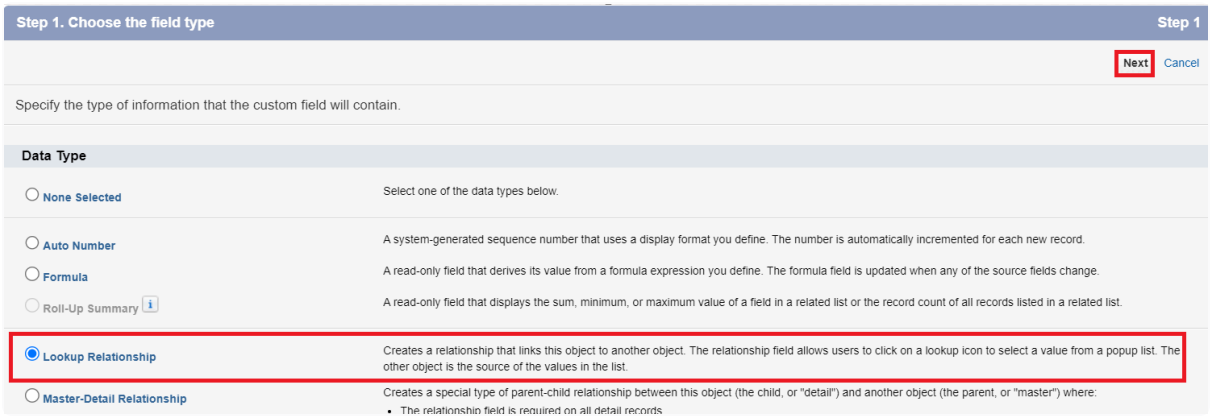
To create fields in an object:

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

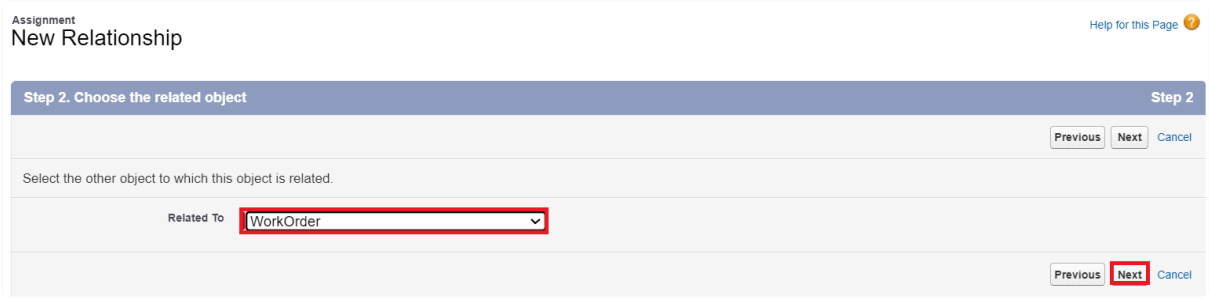
2. Now click on “Fields & Relationships” --> New



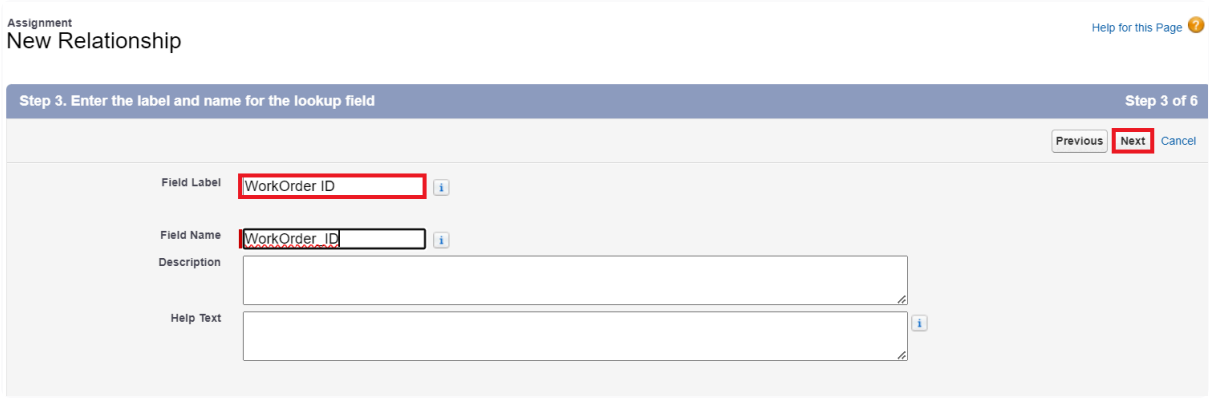
3. Select Data type as “Lookup”.



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.



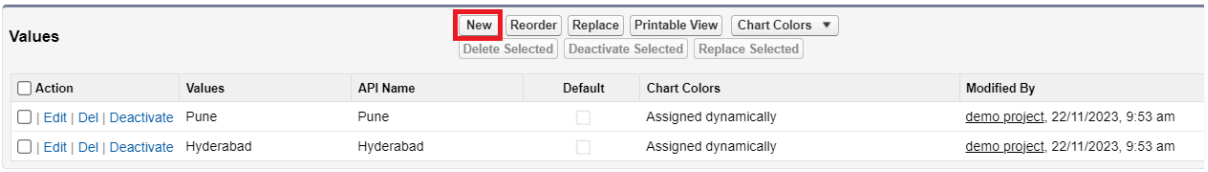
6. Give Field Label as “WorkOrder ID” and click Next.



Next --> Next --> Save & New

Task 2: Manage your picklist values

1. From the setup page go to object manager
2. Search and Select WorkOrder object.
3. Go to fields & relationship, select Location field, scroll down to values and click “New”.



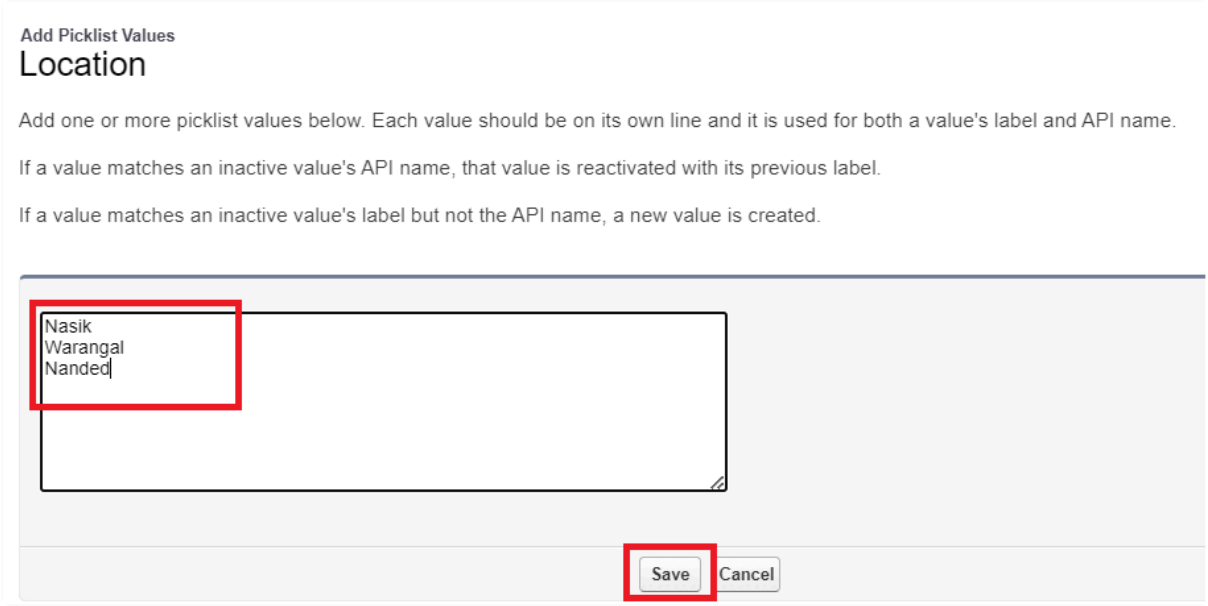
4. Add the below values:

   Nasik

   Warangal

   Nanded

5. Click Save.

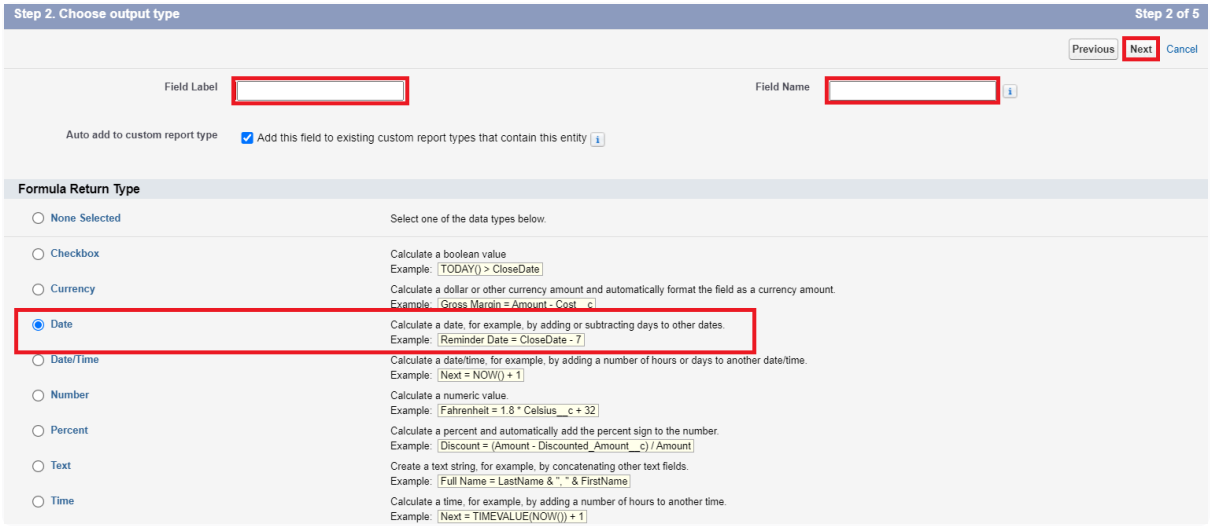


Add following values to the respective fields in WorkOrder object:

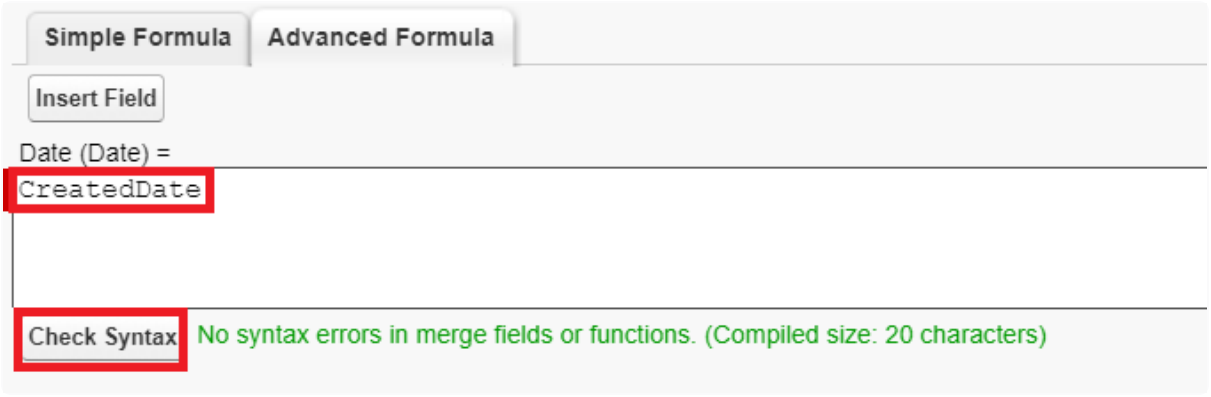
Field	Values
Priority	High
Service Type	Hardware repair Troubleshoot/Debugging Lane-Management

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.



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



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

Task 4: Creating Remaining fields for the respective objects

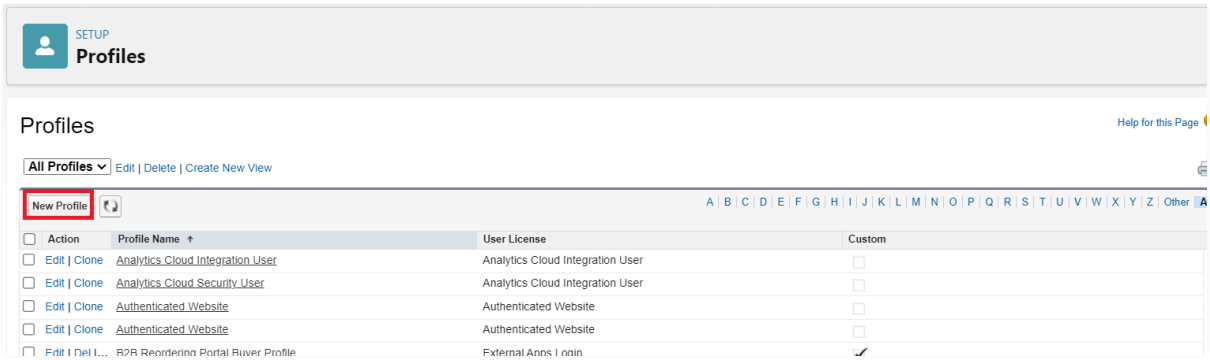
Sl No	Object Name	Field	
		Field Name	**Datatype**
1	Assignment	• Technician ID • Assignment Date • Completion Date	Lookup(Technician) Formula: return type : Date (WorkOrder_ID__r.Date__c) Formula: return type : Date IF(ISPICKVAL(WorkOrder_ID__r.Status__c , 'Resolved'),

Milestone 6- Profiles

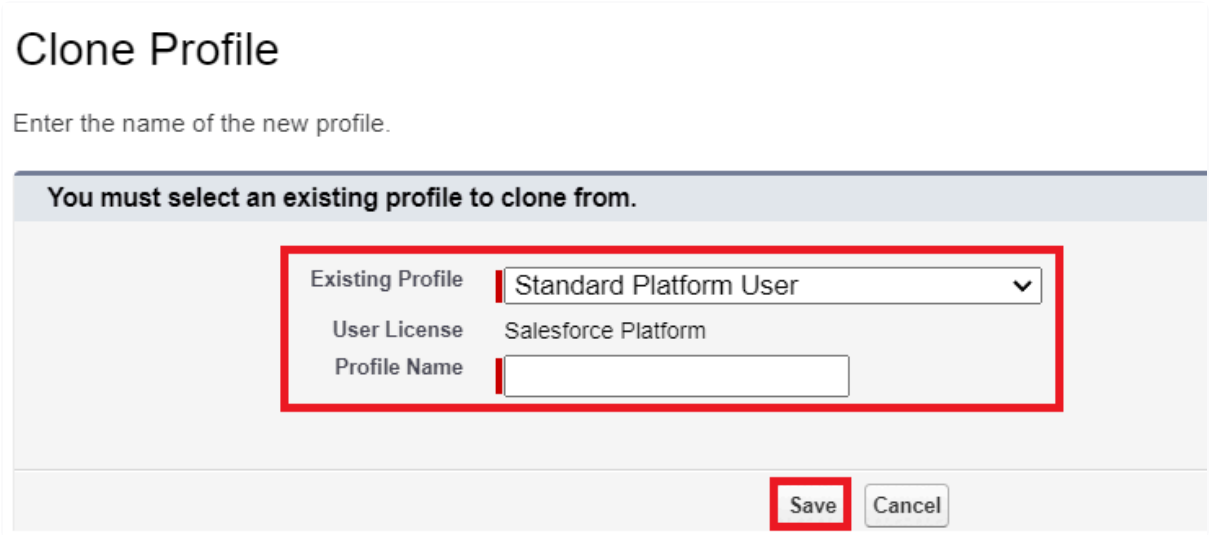
Profile defines what an user is able to do or see in the Salesforce Org

Technician Profile

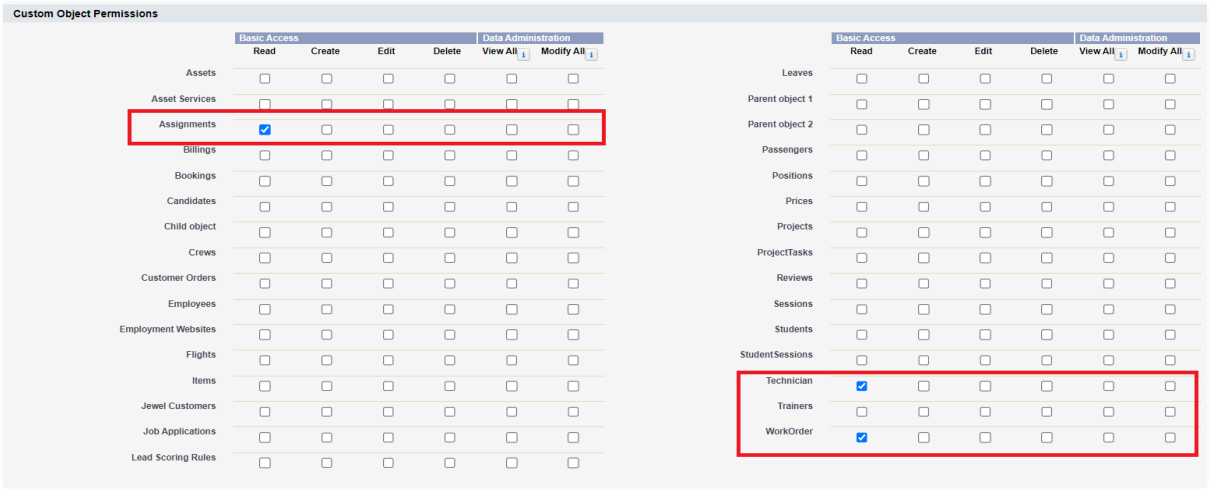
1. Go to setup --> type profiles in quick find box --> click on profiles --> click on new profile.



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

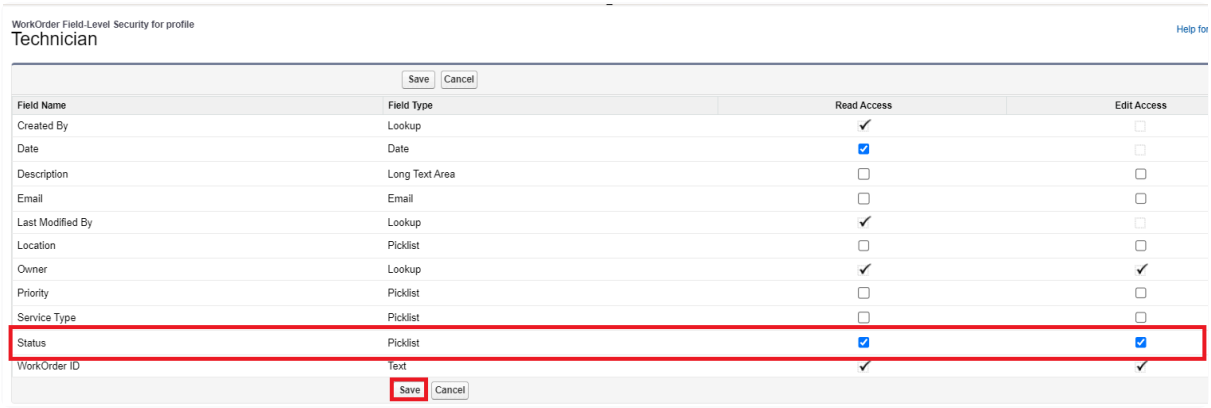


3. While still on the profile page, then click Edit.
4. Scroll down to Custom Object Permissions and Give Read only access permissions for Technician, WorkOrder and Assignment objects and field access permission as shown below:



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.

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



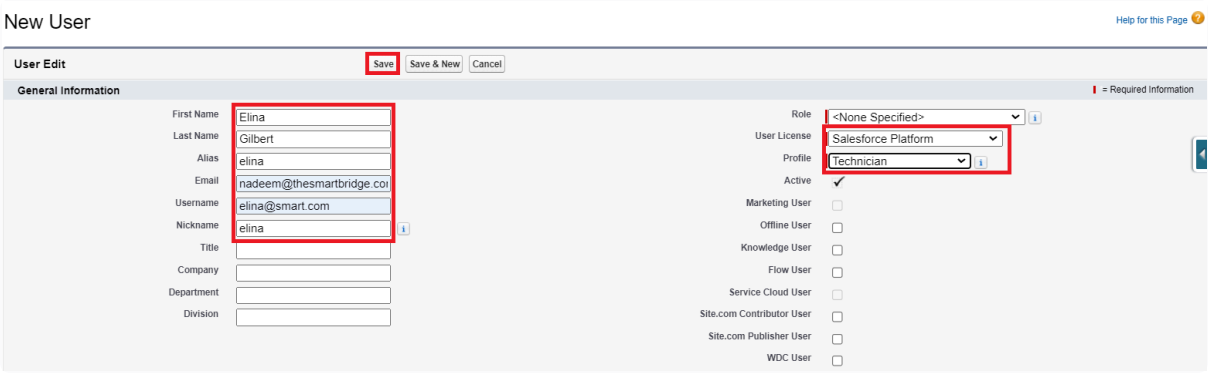
8. Click on Save.

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

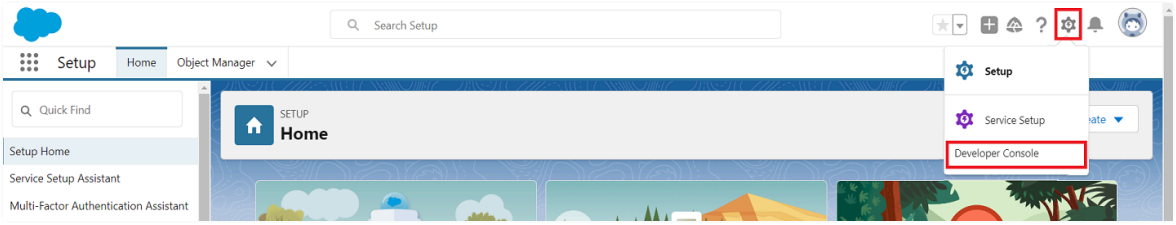


3. Save.

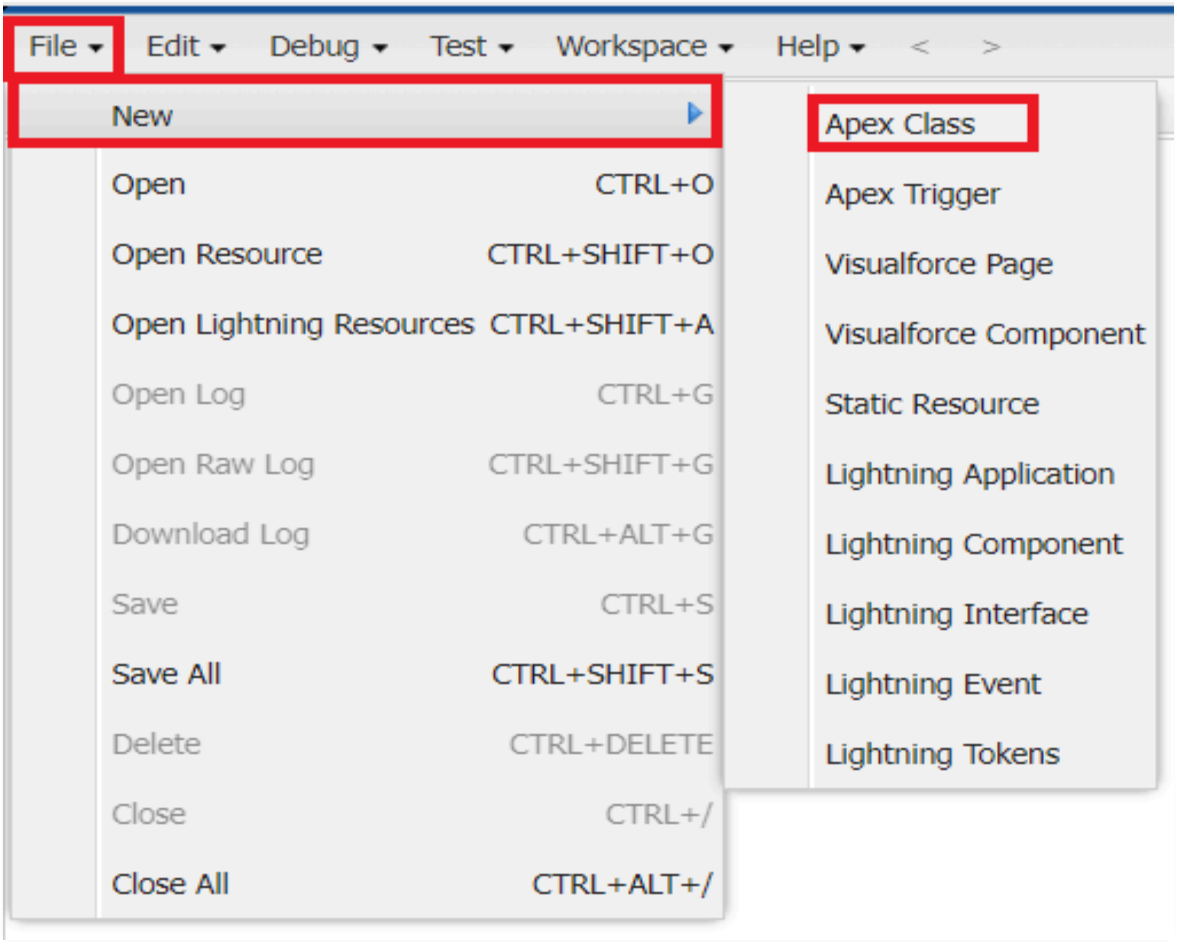
Milestone-8: Apex Trigger

Create an Apex Class

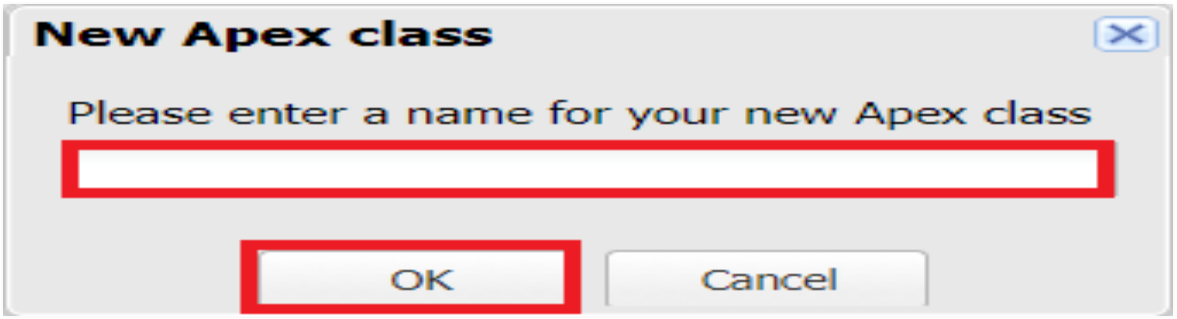
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



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

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> techniciantoAssignment = 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, Availibility__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);
```

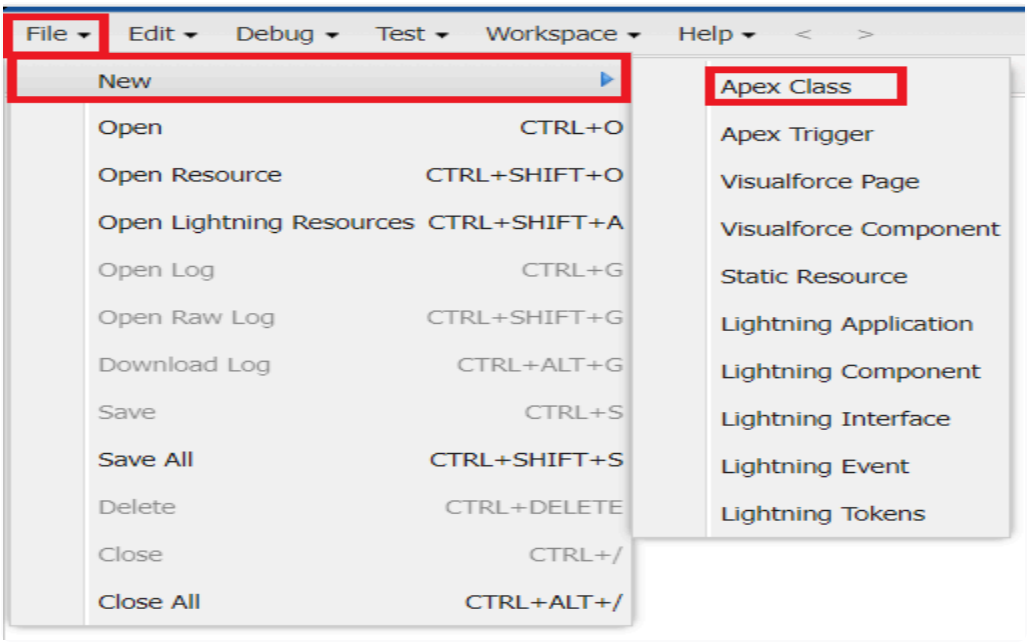
```
            system.debug('error 1 ----> the maptotech is empty ---> ' +
maptotech.get(num2));
            if(valofMap.contains(t.Skills__c) && ValofMap.contains(t.Location__c) &&
t.Availibility__c == 'Available'){
                techid.put(num2,t.Id);
            }
            }

        }
        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;
        }
    }
}
```

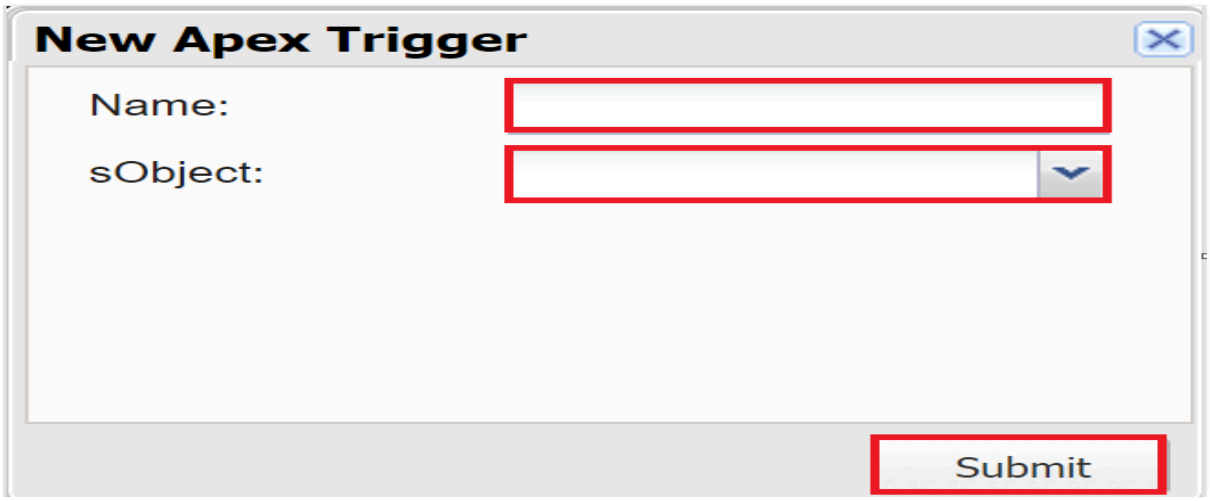
Save the code.(click on file --> Save)

Create an Apex Trigger

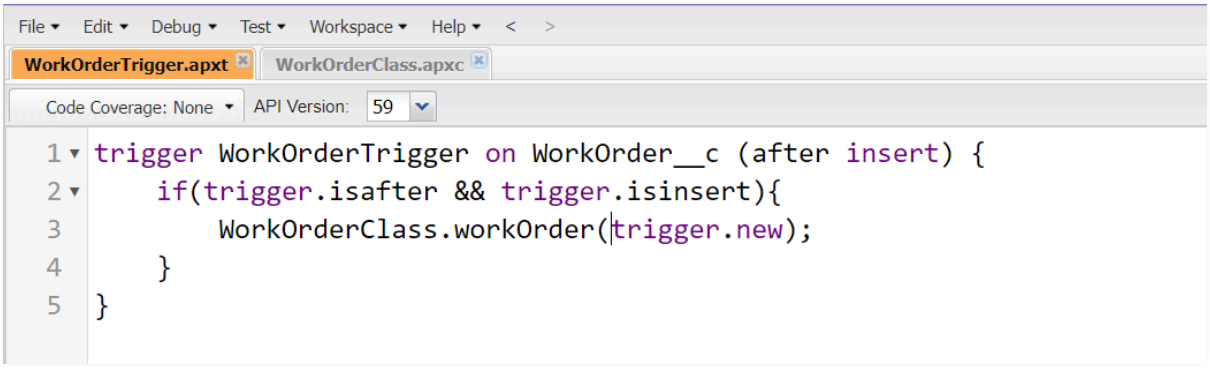
1. To create a new Apex Class follow the below steps:
   Click on the file --> New --> Apex Class.



2. Give the Apex Trigger name as “WorkOrderTrigger”, and select “WorkOrder__c” from the dropdown for sObject.



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



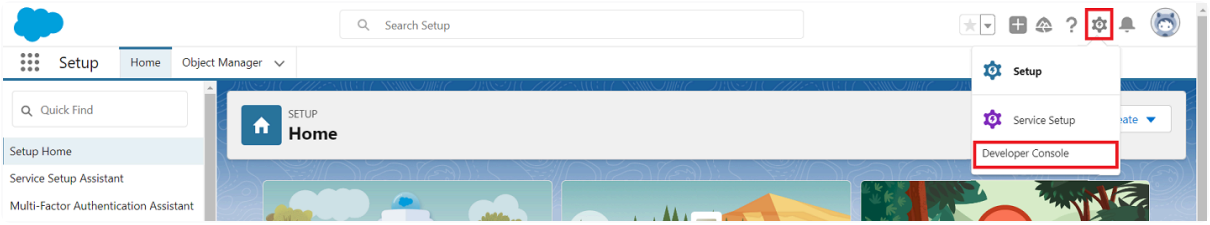
Source Code:

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

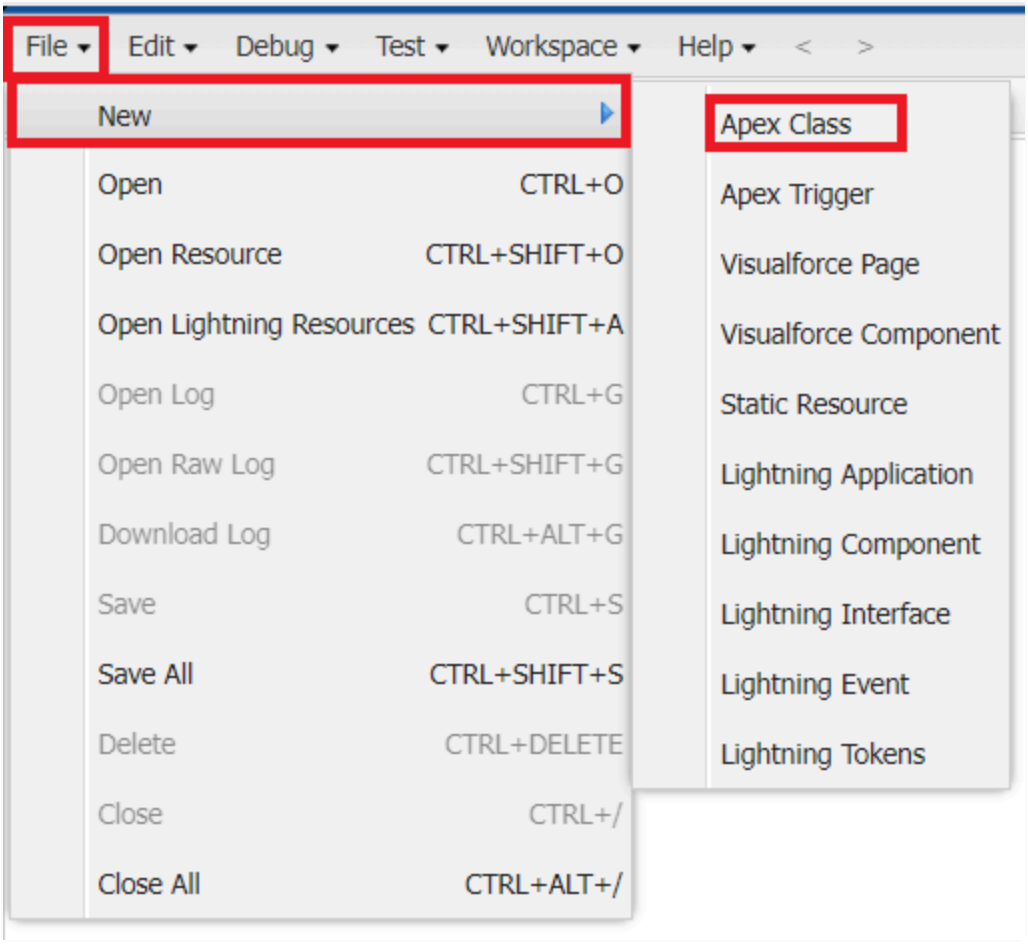
5. Save the code.(click on file --> Save)

Create an Apex Class

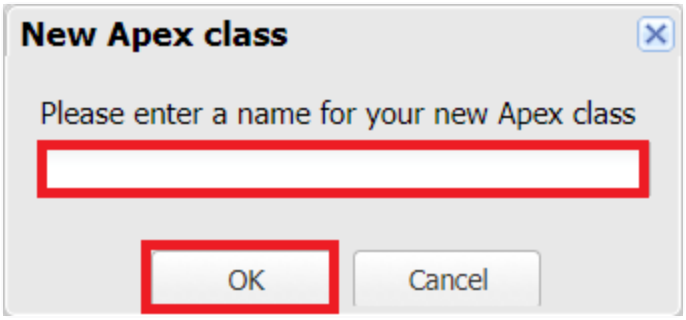
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



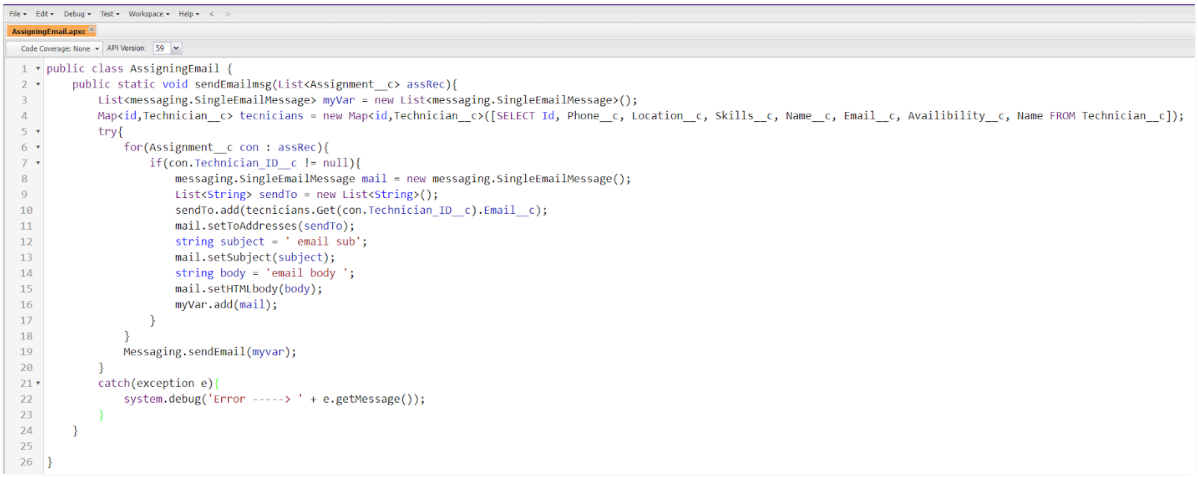
2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3. To create a new Apex Class follow the below steps:
   Click on the file --> New --> Apex Class.



4. Give the Apex Class name as “AssigningEmail”.



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



```
public class AssigningEmail {
    public static void sendEmailmsg(List<Assignment__c> assRec){
        List<messaging.SingleEmailMessage> myVar = new List<messaging.SingleEmailMessage>();
        Map<id,Technician__c> tecnicians = new Map<id,Technician__c>([SELECT Id, Phone__c, Location__c, Skills__c, Name__c, Email__c, Availibility__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(tecnicians.Get(con.Technician_ID__c).Email__c);
                    mail.setToAddresses(sendTo);
                    string subject = ' email sub';
                    mail.setSubject(subject);
                    string body = 'email body ';
                    mail.setHTMLbody(body);
                    myVar.add(mail);
                }
            }
            Messaging.sendEmail(myvar);
        }
        catch(exception e){
            system.debug('Error -----> ' + e.getMessage());
        }
    }
}
```

Source Code:

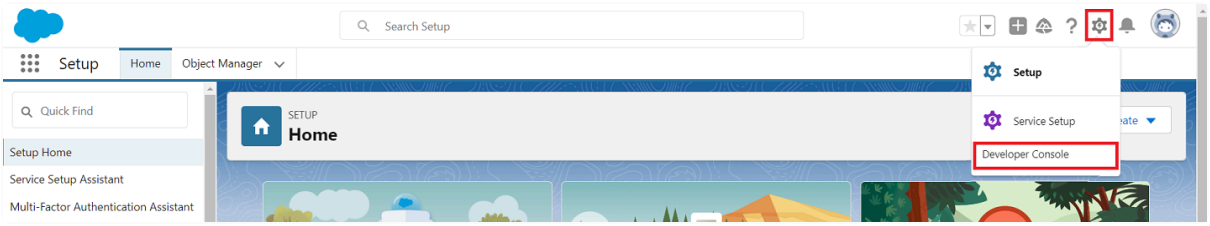
```
public class AssigningEmail {
   public static void sendEmailmsg(List<Assignment__c> assRec){
      List<messaging.SingleEmailMessage> myVar = new
List<messaging.SingleEmailMessage>();
      Map<id,Technician__c> tecnicians = new Map<id,Technician__c>([SELECT Id,
Phone__c, Location__c, Skills__c, Name__c, Email__c, Availibility__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(tecnicians.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());
        }
    }
}
```

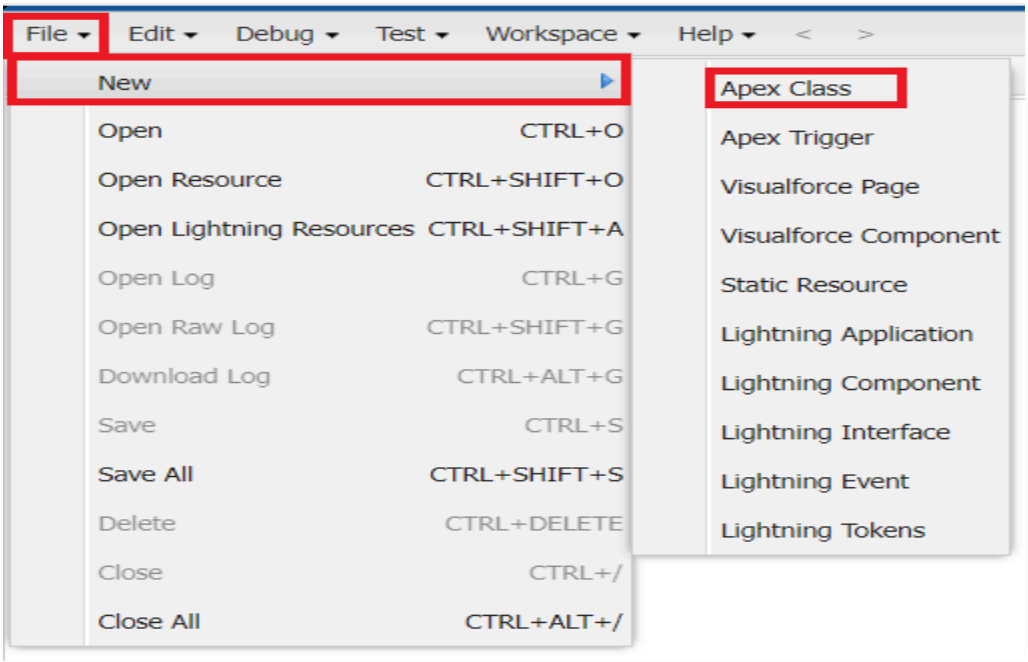
7. Save the code.(click on file --> Save)

Create an Apex Class

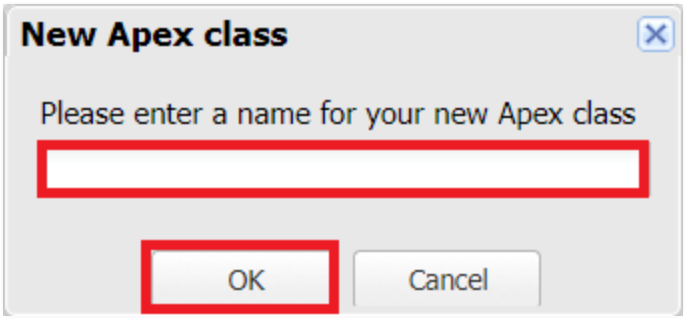
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



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

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> tecnicians = new Map<id,Technician__c>([SELECT Id,
Phone__c, Location__c, Skills__c, Name__c, Email__c, Availibility__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(tecnicians.Get(con.Technician_ID__c).Email__c);
                    mail.setToAddresses(sendTo);
```

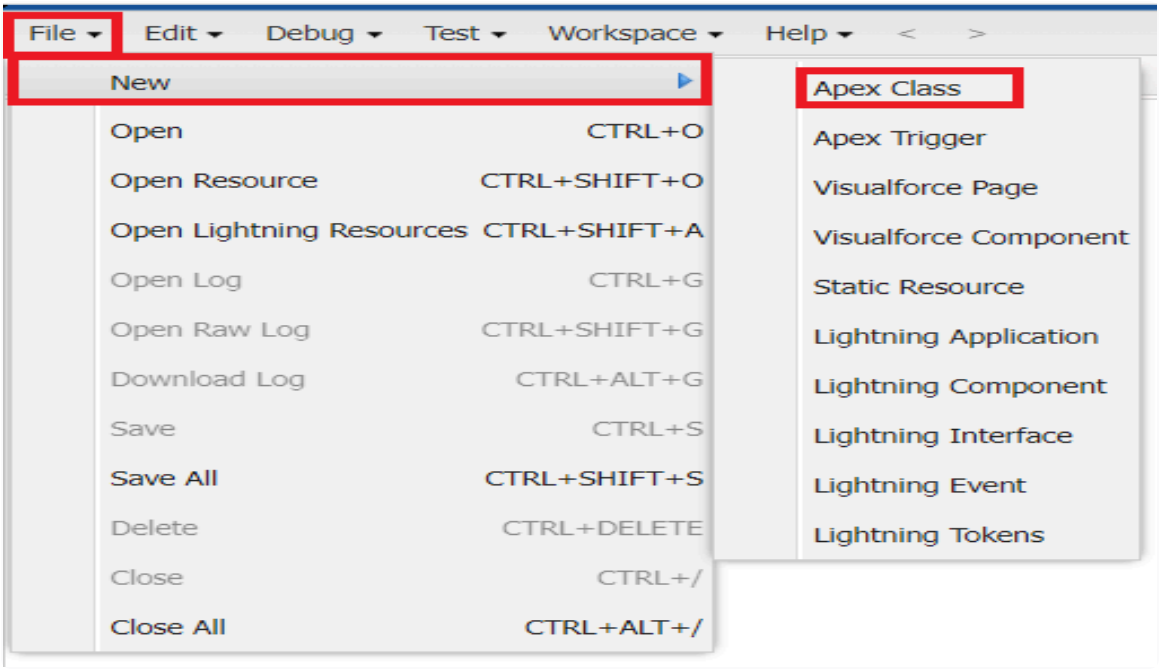
```
                string subject = 'WorkOrder Assignment ';
                mail.setSubject(subject);
                string body = 'The following WorkOrder has been assigned to you ';
                mail.setHTMLbody(body);
                myVar.add(mail);
            }
          }
          Messaging.sendEmail(myvar);
        }
        catch(exception e){
          system.debug('Error -----> ' + e.getMessage());
        }
      }
    }
```

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

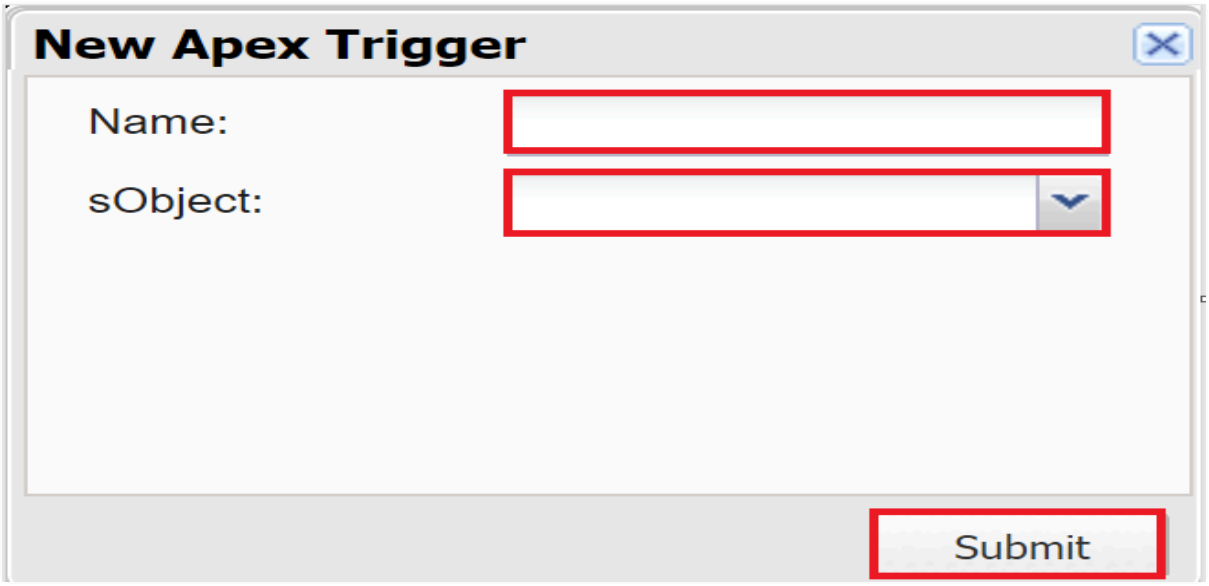
Create an Apex Trigger

To create a new Apex Class follow the below steps:

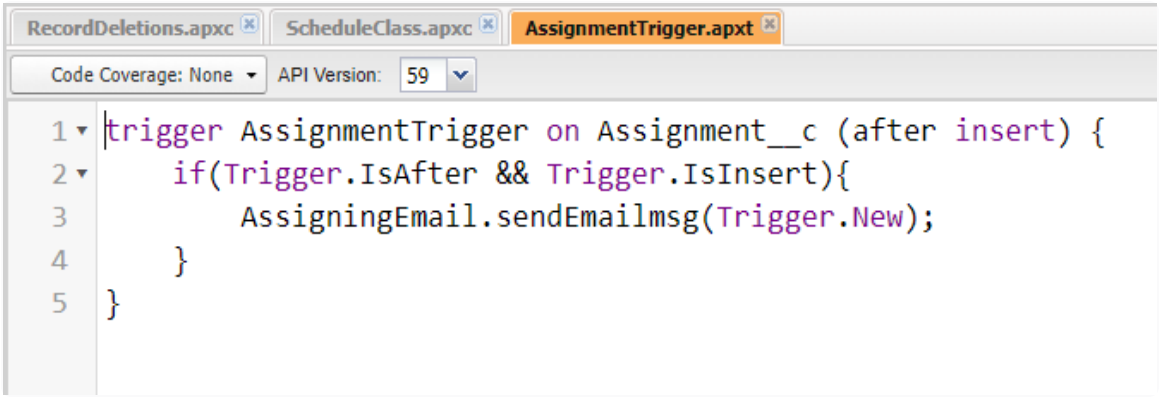
1. Click on the file --> New --> Apex Class.



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



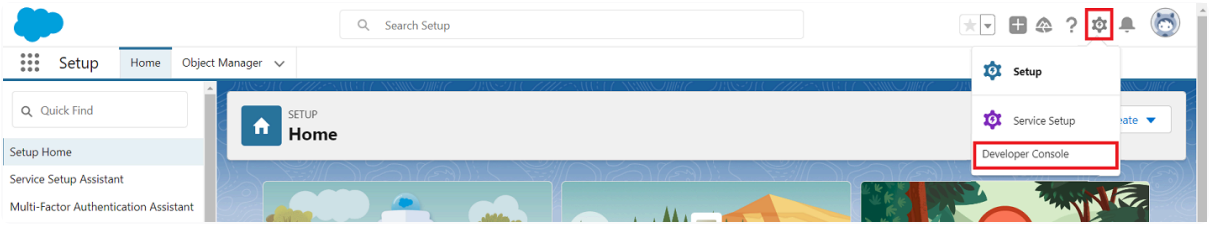
Source Code:

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

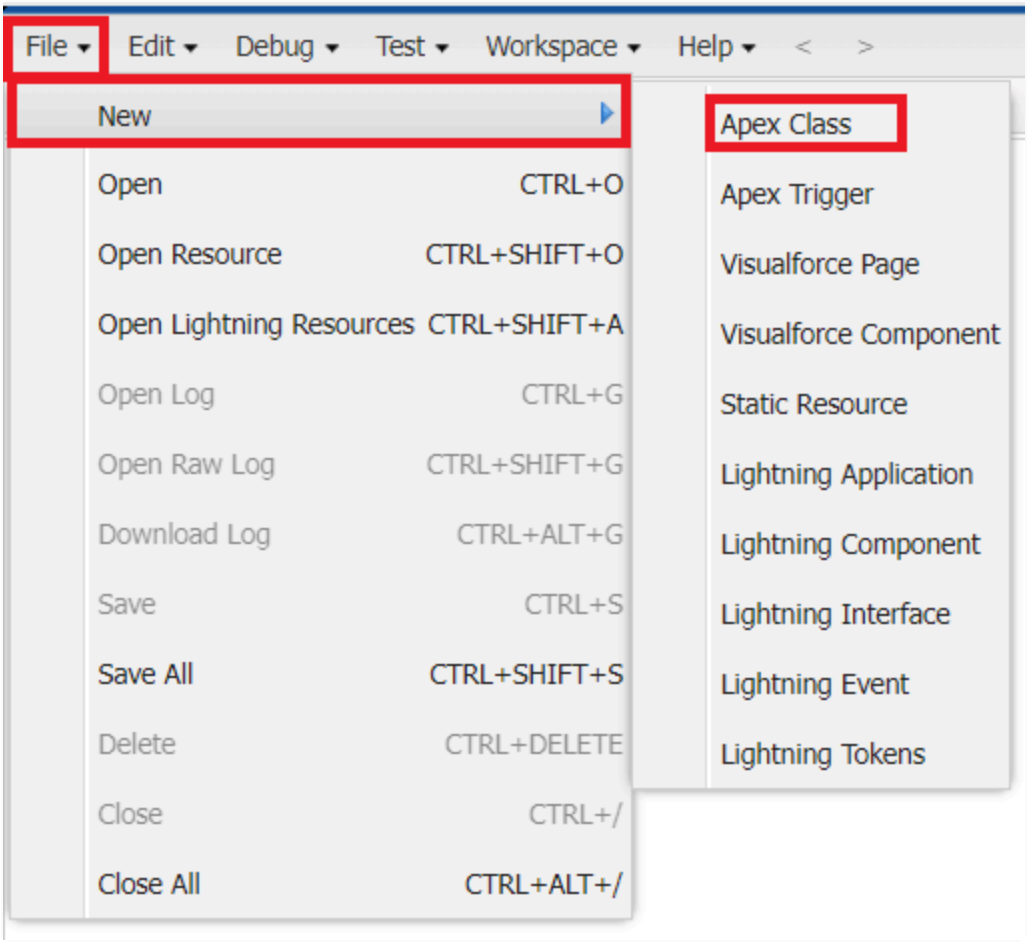
Save the code.(click on file --> Save)

Create an Apex Class

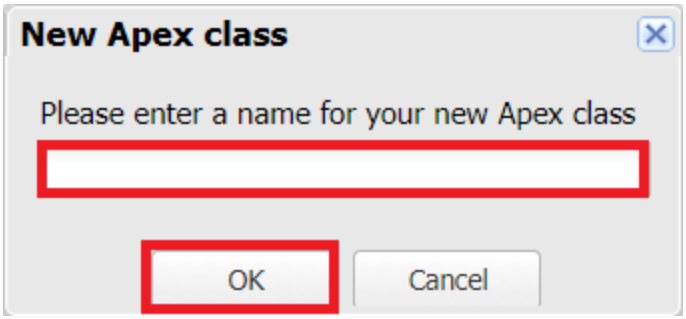
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



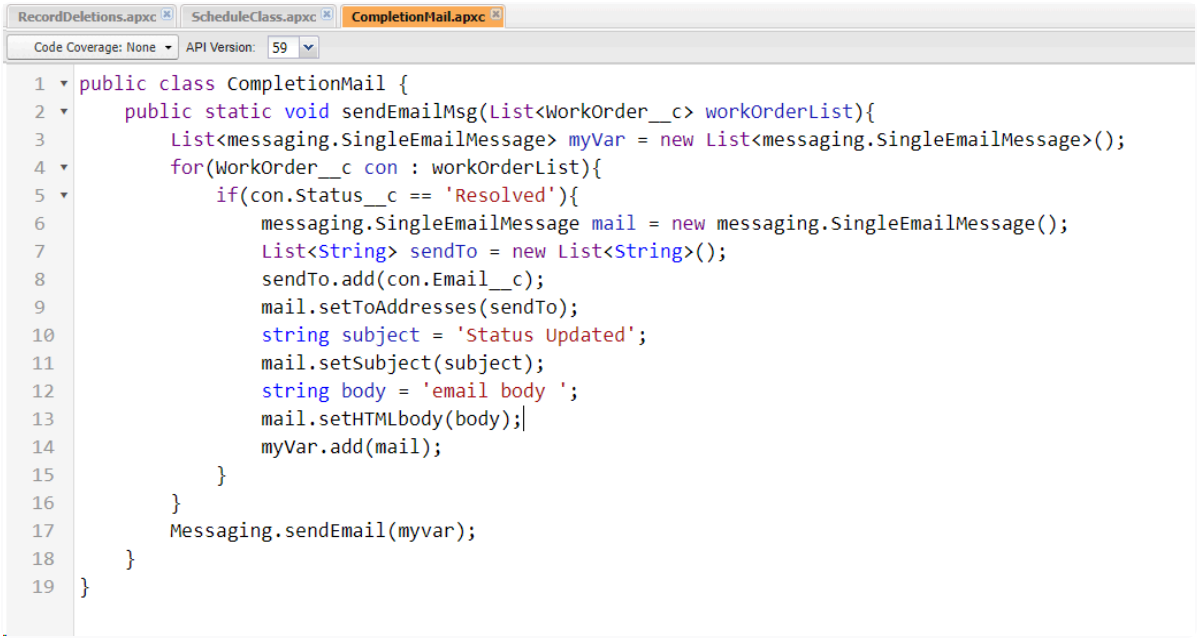
2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3. To create a new Apex Class follow the below steps:
   Click on the file --> New --> Apex Class.



4. Give the Apex Class name as “CompletionMail”.



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



Source Code:

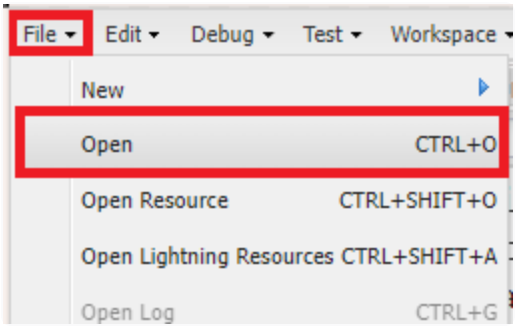
```
public class CompletionMail {
   public static void sendEmailMsg(List<WorkOrder__c> workOrderList){
      List<messaging.SingleEmailMessage> myVar = new
List<messaging.SingleEmailMessage>();
      for(WorkOrder__c con : workOrderList){
         if(con.Status__c == 'Resolved'){
            messaging.SingleEmailMessage 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);
  }
}
```

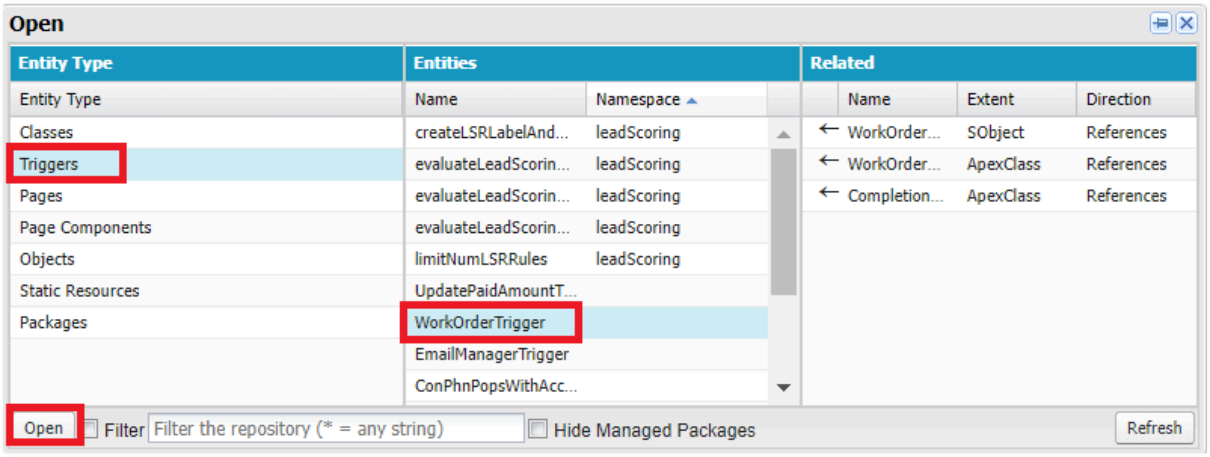
7.
8. Save the code.(click on file --> Save)

Create an Apex Trigger

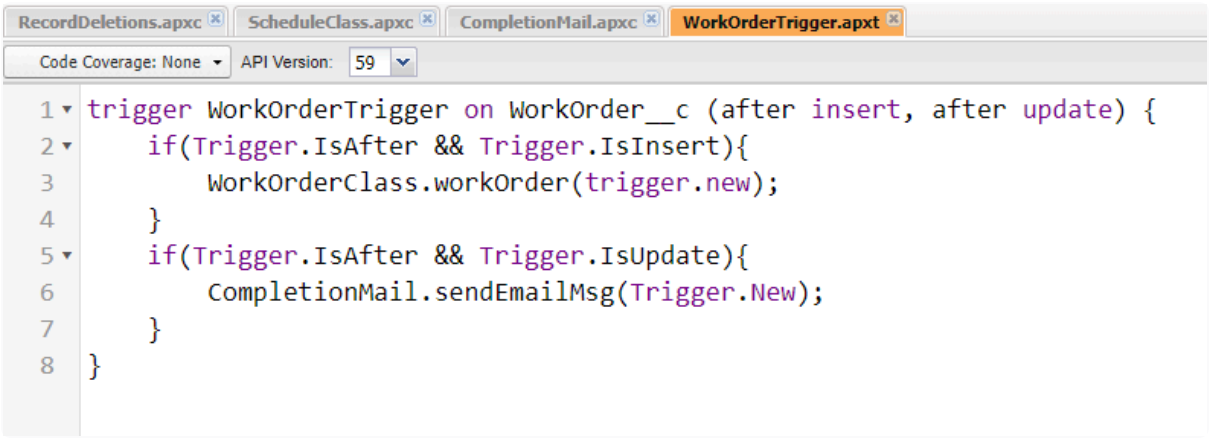
1. Click on the file --> Open.



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



3. Now write the code logic here.



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)

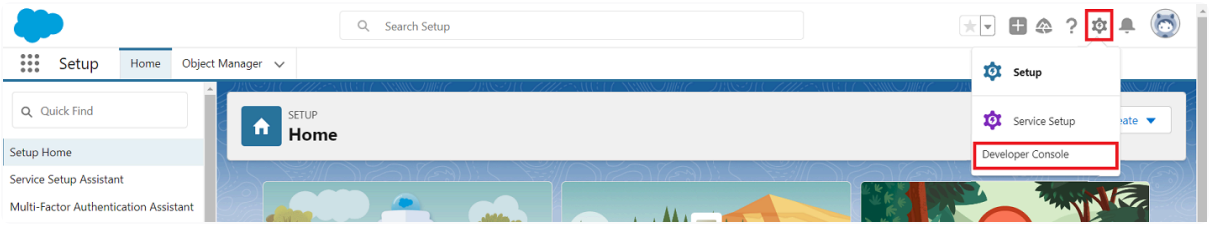
Create an Asynchronous Apex Class

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

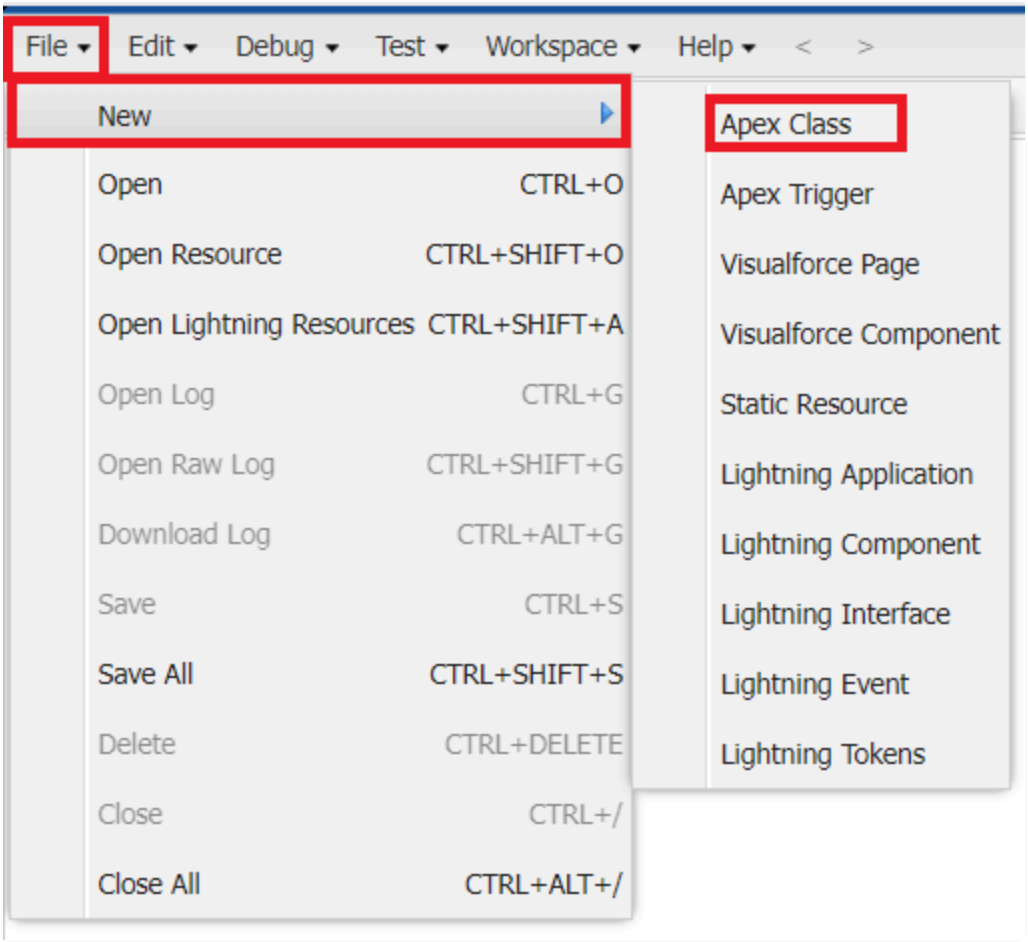
1. Completed date should be more than 30 days.
2. Status should be 'Resolved'.

Create an Apex Class

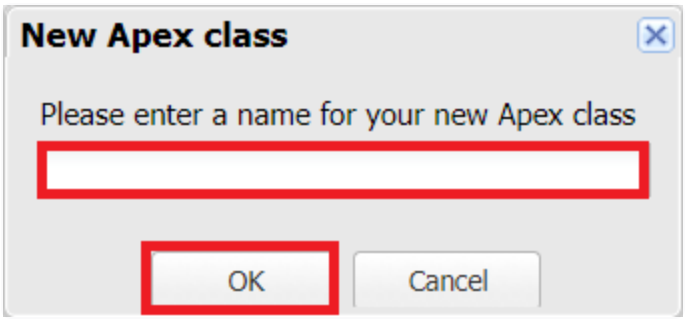
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



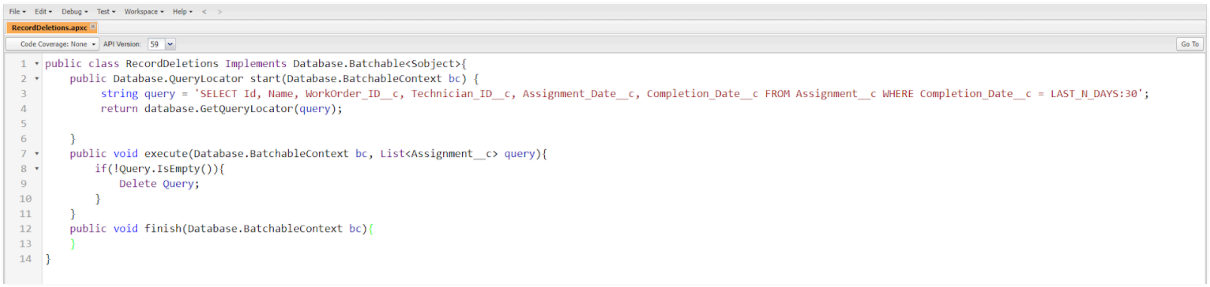
2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3. To create a new Apex Class follow the below steps:
   Click on the file --> New --> Apex Class.



4. Give the Apex Class name as “RecordDeletion”.



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



```
public class RecordDeletions Implements Database.Batchable<Sobject>{
    public Database.QueryLocator start(Database.BatchableContext bc) {
         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';
         return database.GetQueryLocator(query);

    }
    public void execute(Database.BatchableContext bc, List<Assignment__c> query){
        if(!Query.IsEmpty()){
            Delete Query;
        }
    }
    public void finish(Database.BatchableContext bc){
    }
}
```

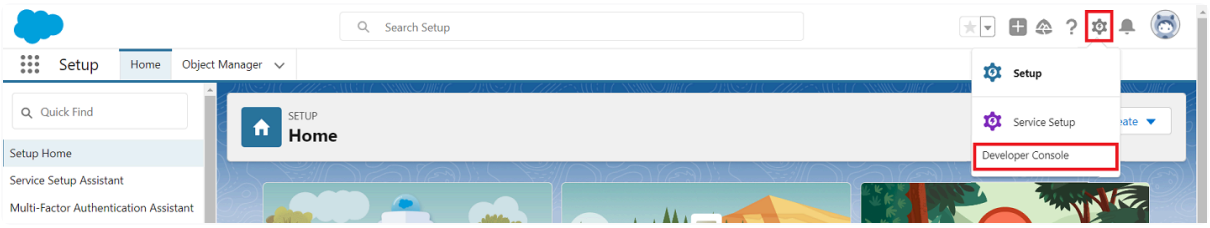
Source Code:

```
public class RecordDeletions Implements Database.Batchable<Sobject>{
    public Database.QueryLocator start(Database.BatchableContext bc) {
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';
        return database.GetQueryLocator(query);
    }
    public void execute(Database.BatchableContext bc, List<Assignment__c> query){
        if(!Query.IsEmpty()){
            Delete Query;
        }
    }
    public void finish(Database.BatchableContext bc){
    }
}

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

Create an Apex Schedule Class

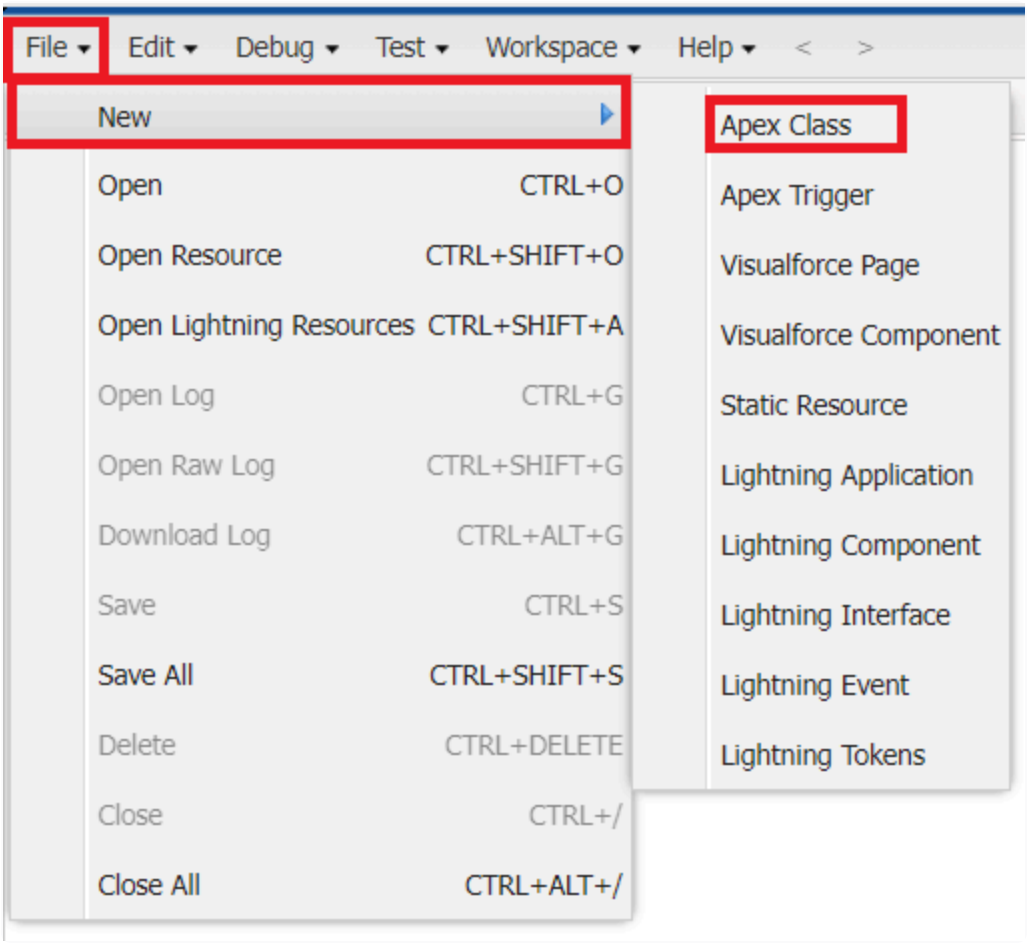
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



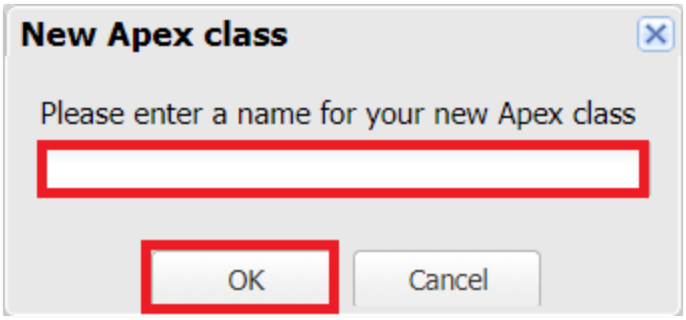
2. Then we can see the Developer console. Click on the developer console and you

will navigate to a new console window.

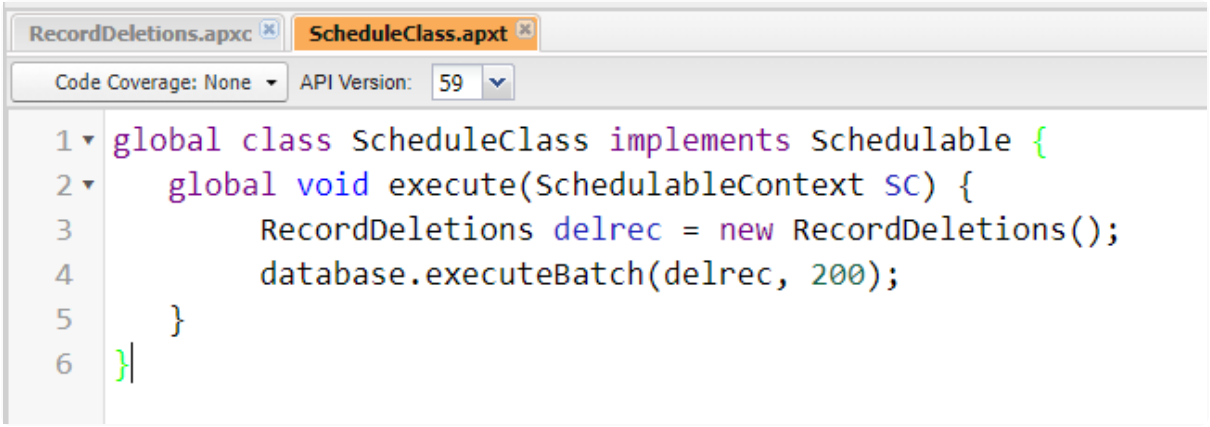
3. To create a new Apex Class follow the below steps:
   Click on the file --> New --> Apex Class.



4. Give the Apex Class name as “ScheduleClass”.



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



Source Code:

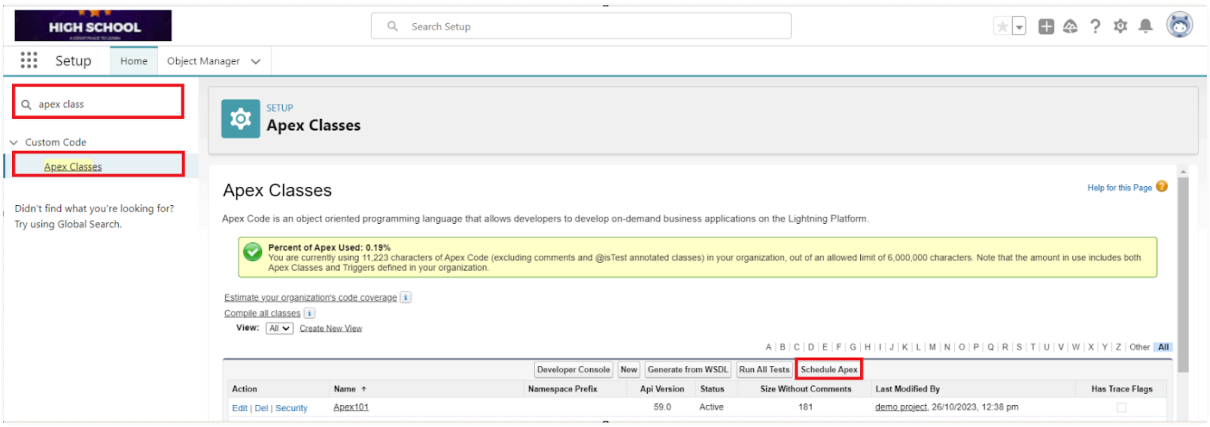
```
global class ScheduleClass implements Schedulable {
  global void execute(SchedulableContext SC) {
      RecordDeletions delrec = new RecordDeletions();
      database.executeBatch(delrec, 200);
  }
}
```

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

Create a Schedule Apex

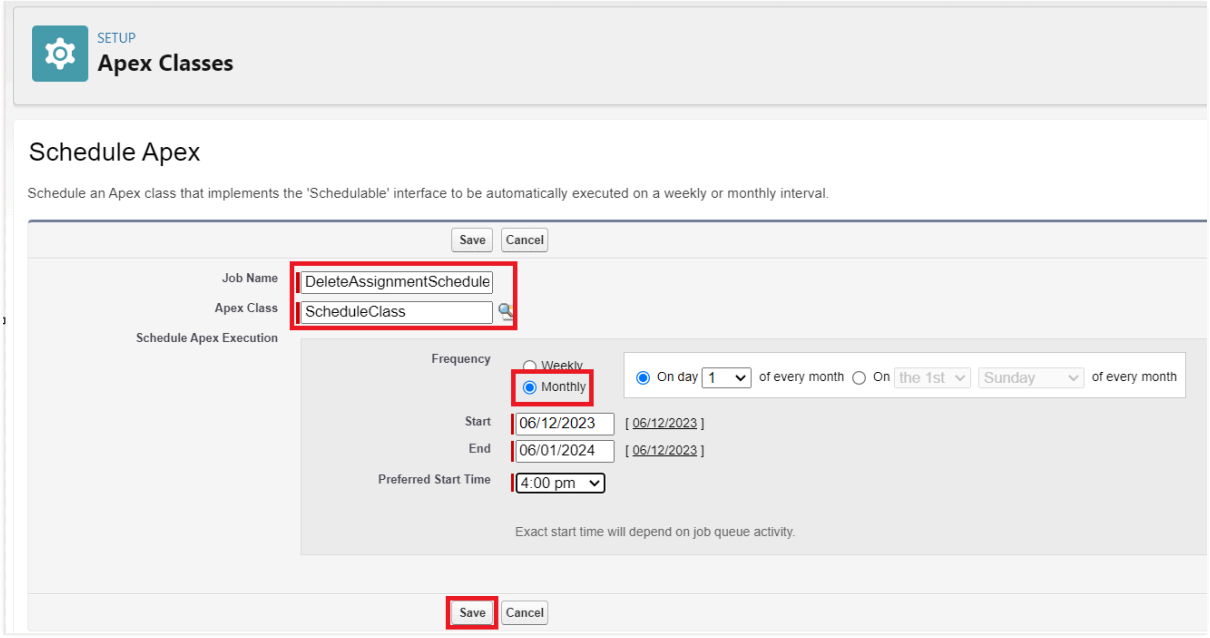
Schedule the Apex class:

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



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

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



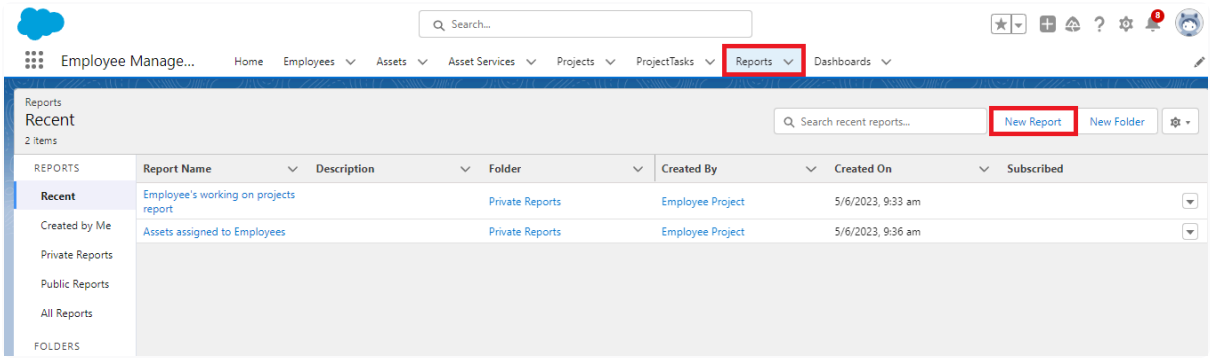
4. Click Save.

Milestone-9: Reports & Dashboards

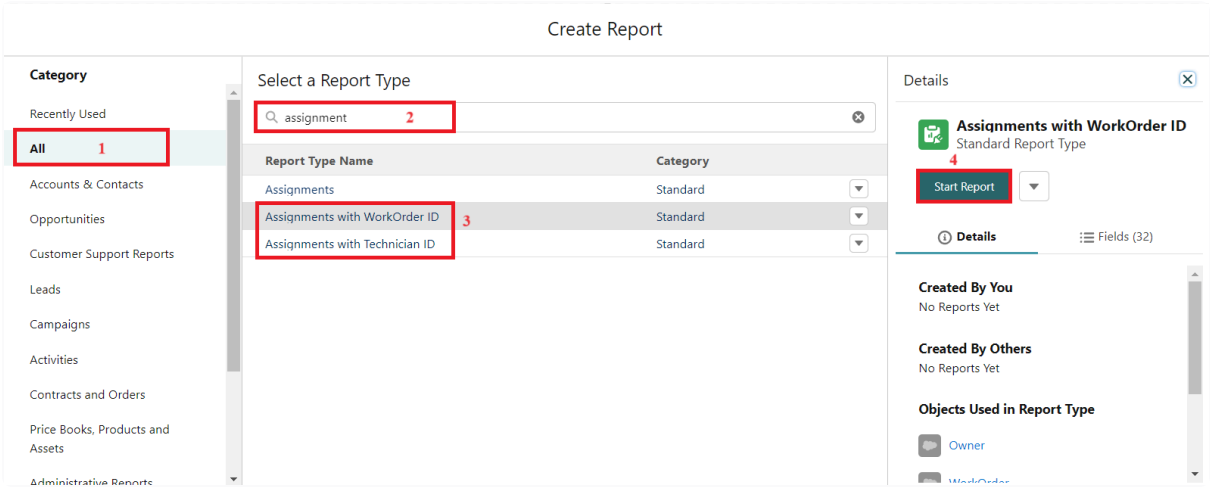
Salesforce Reports and Dashboards are powerful tools that empower users to visualize and analyze data within the Salesforce platform. They play a crucial role in providing insights, monitoring performance, and making informed business decisions.

Report

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

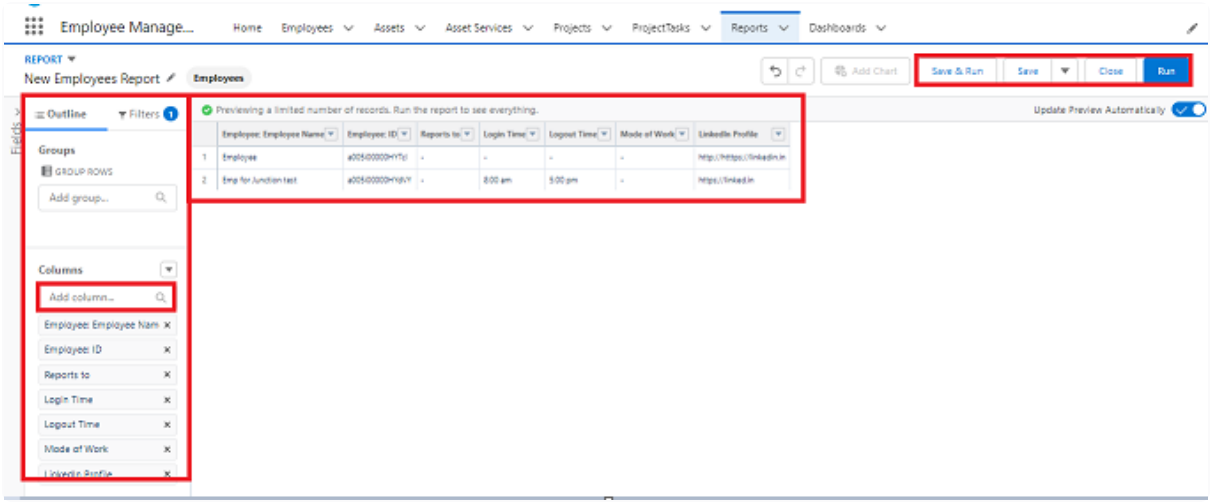


3. Select report type from category or from report type panel or from search panel --> click on start report.



4. Customize your report

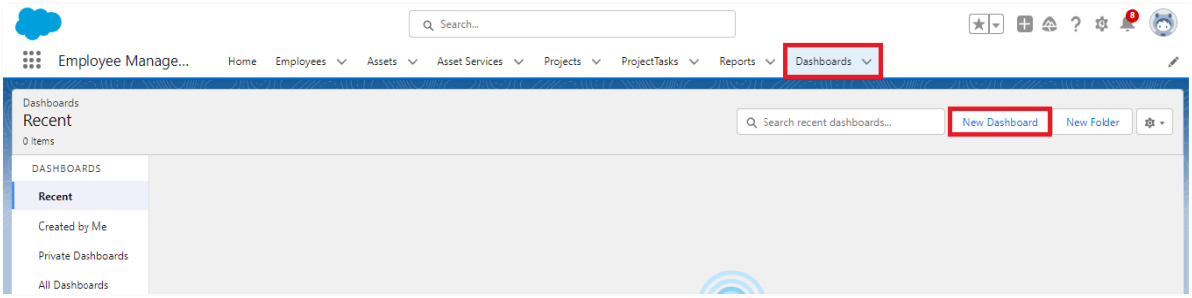
- Add fields from left pane as shown below
- Grouped by workorder ID



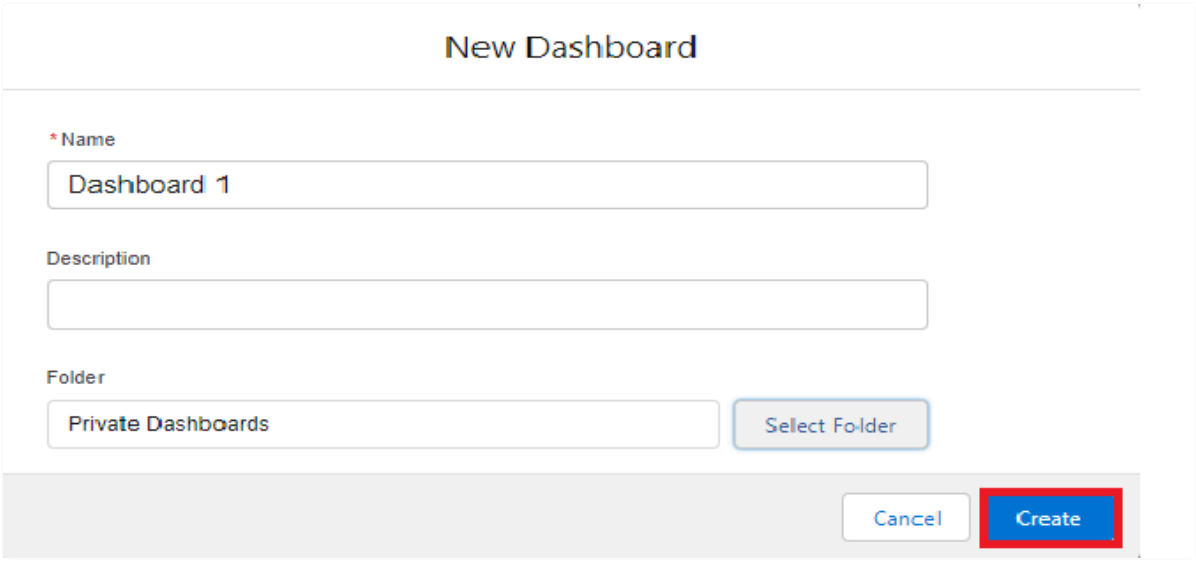
5. Save or run it.
   Note: Reports may get varied from the above pictures as the data might be different.

Dashboard

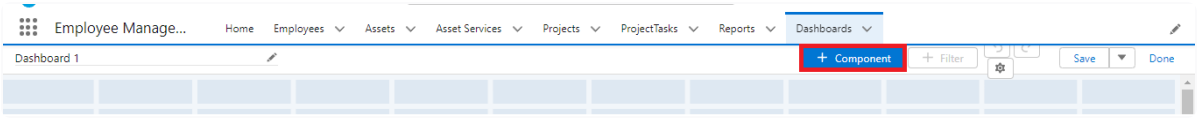
1. Go to the app --> click on the Dashboards tabs.



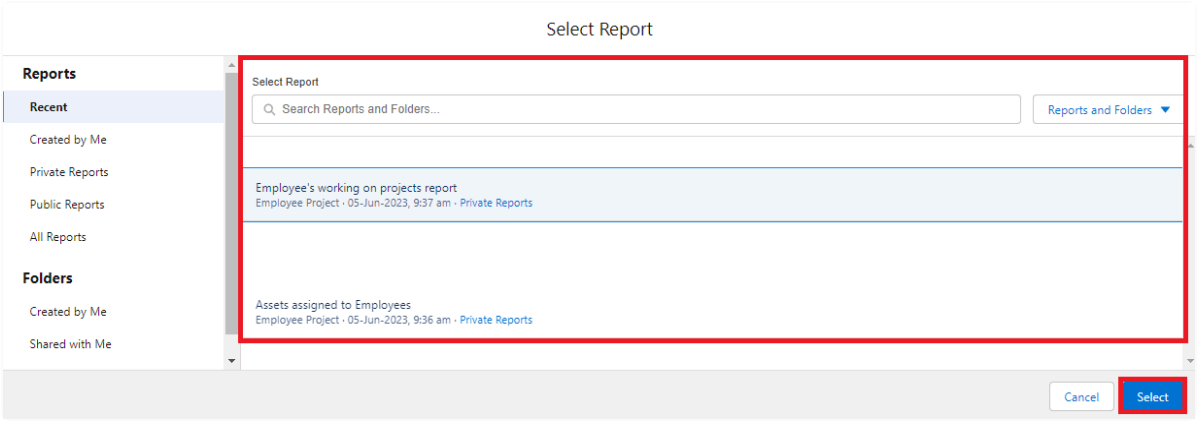
2. Give a Name and click on Create.



3. Select add component.



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



5.