Field service workorder optimization

INTRODUCTION

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 technician’s 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.

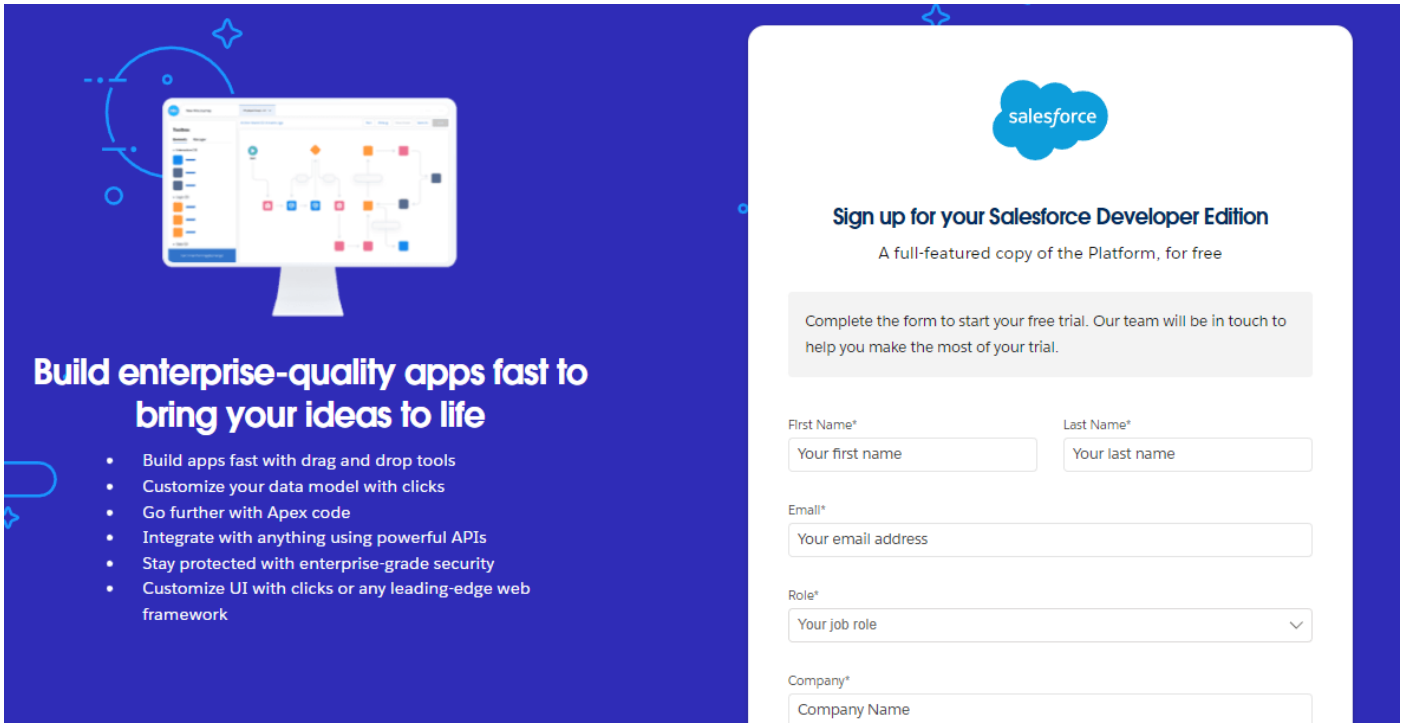
Sales force

Creating a Sales force Developer Edition org allows developers to experiment, innovate, and build customized solutions within a controlled environment. With access to Sales force's powerful development tools and features, developers can prototype, test, and refine their applications, empowering them to deliver robust and tailored solutions to meet unique business requirements. As a Sales force Developer for an organization you must have a Sales force developer edition org in order to do all the required works.

ACTIVITY 1: Creating Developer Account

Creating a developer org in sales force.

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



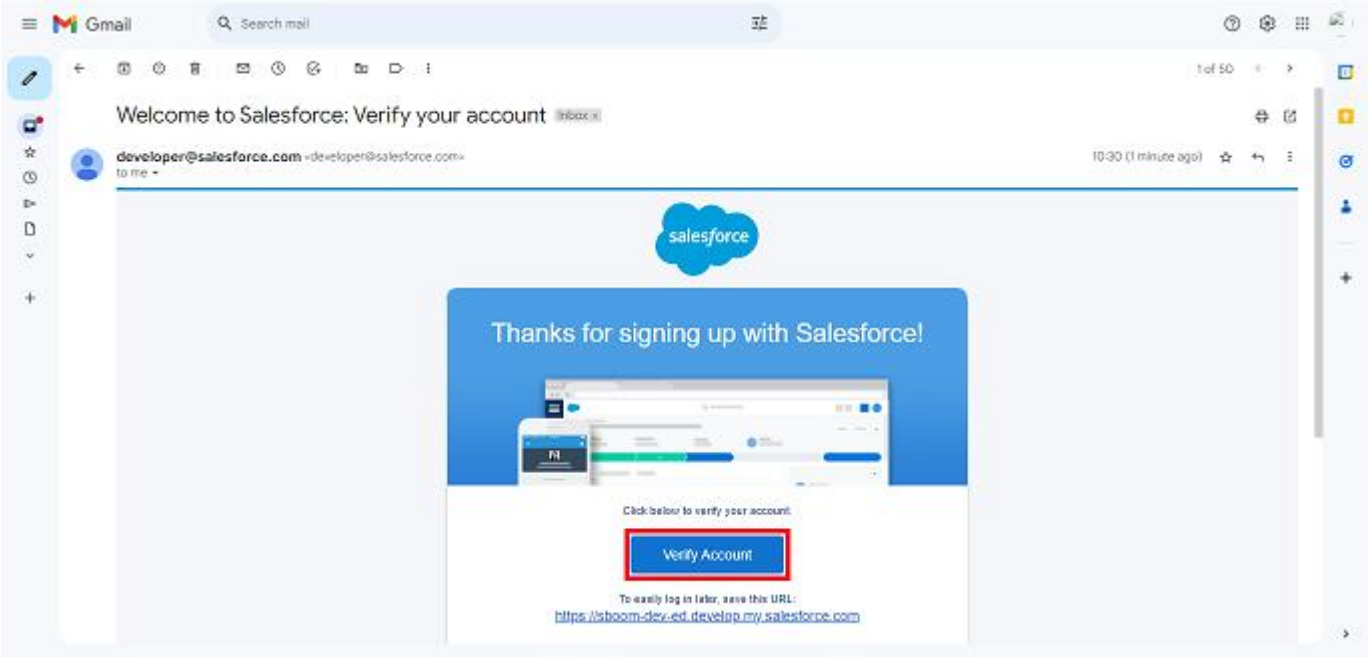
1. First name - SAGAR
2. Last name - ARIGALA
3. Email – Sagararigala@Gmail.Com
4. Role : Developer
5. Company : Gayatri Degree College Tirupati
6. County : India

Field service workorder optimization

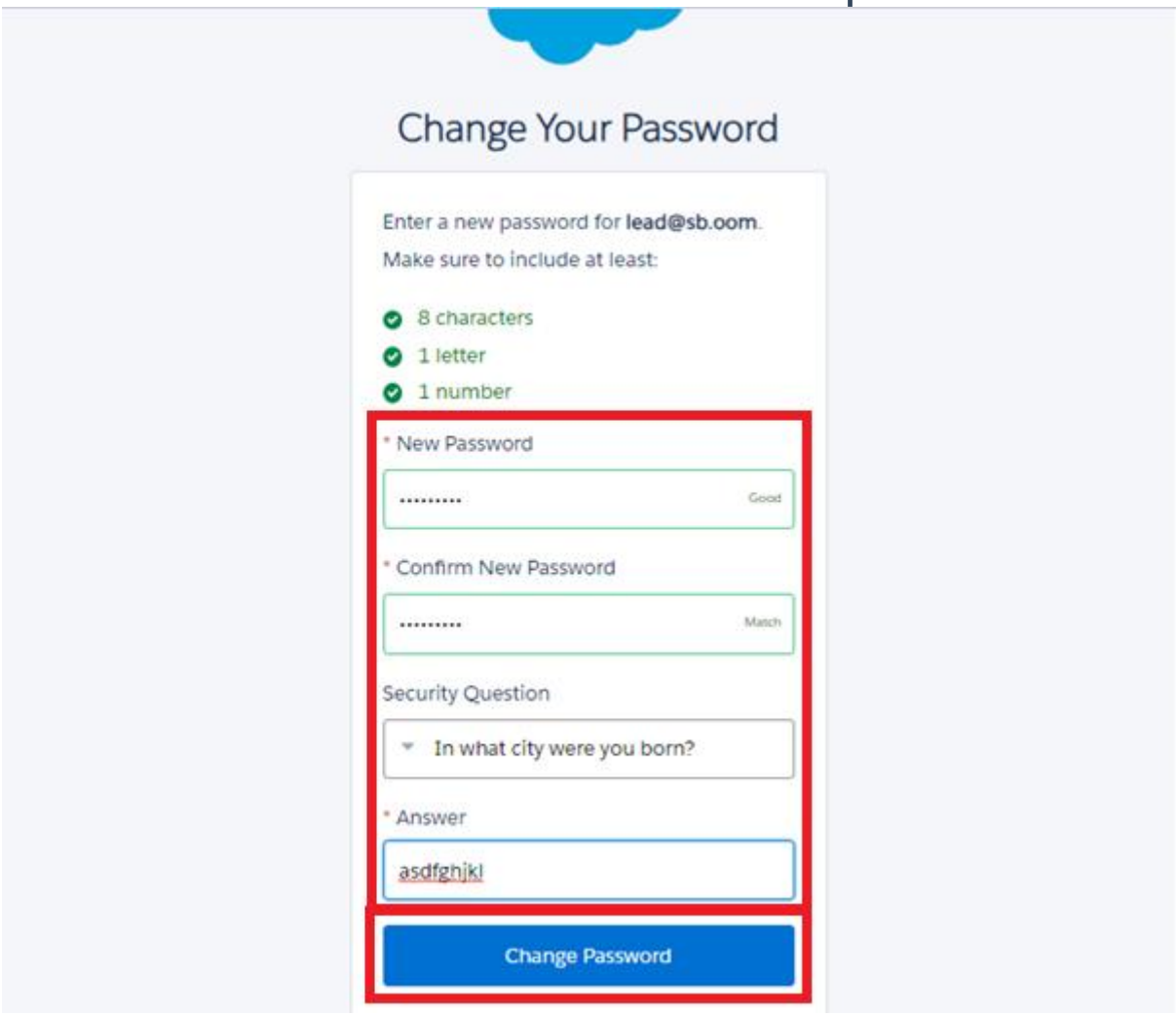
This need not be an actual email id, you can give anything in the format : username@organization.com
Click on sign me up after filling these.

ACTIVITY 2: Account Activation

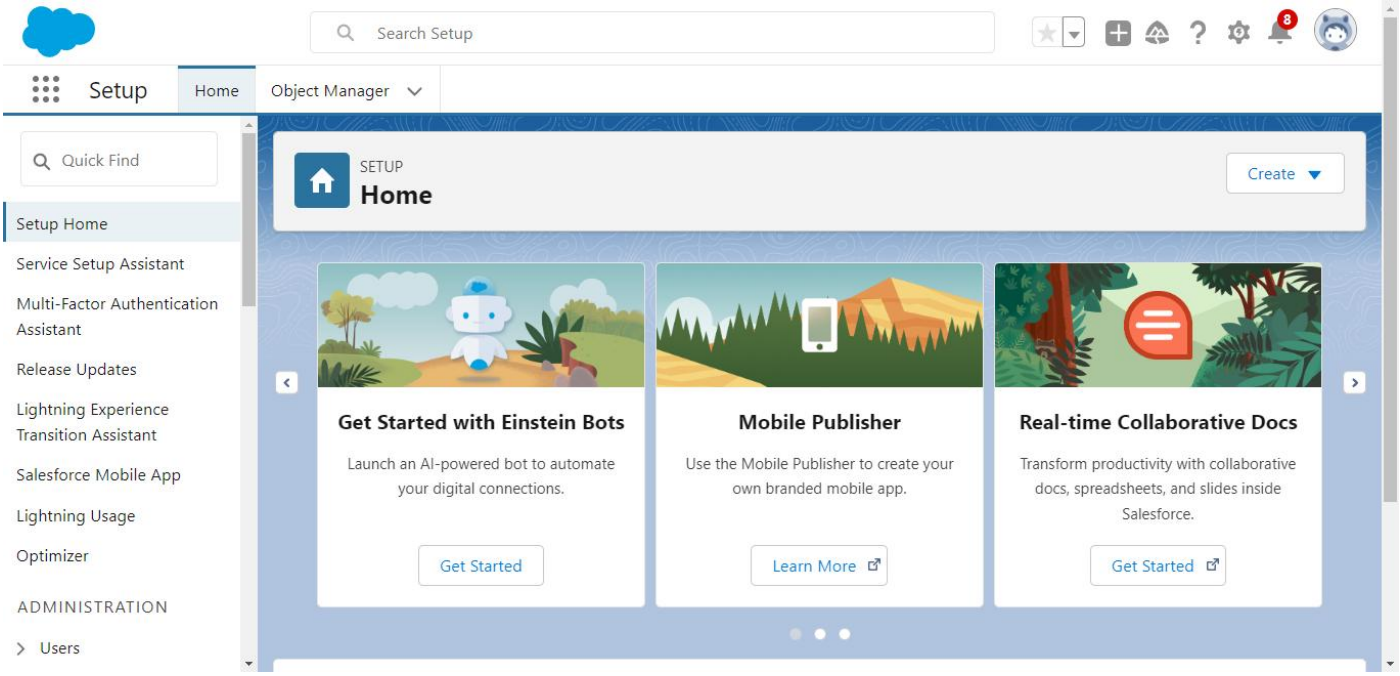
1.Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins.



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 sales force setup page.



Field service workorder optimization

Object

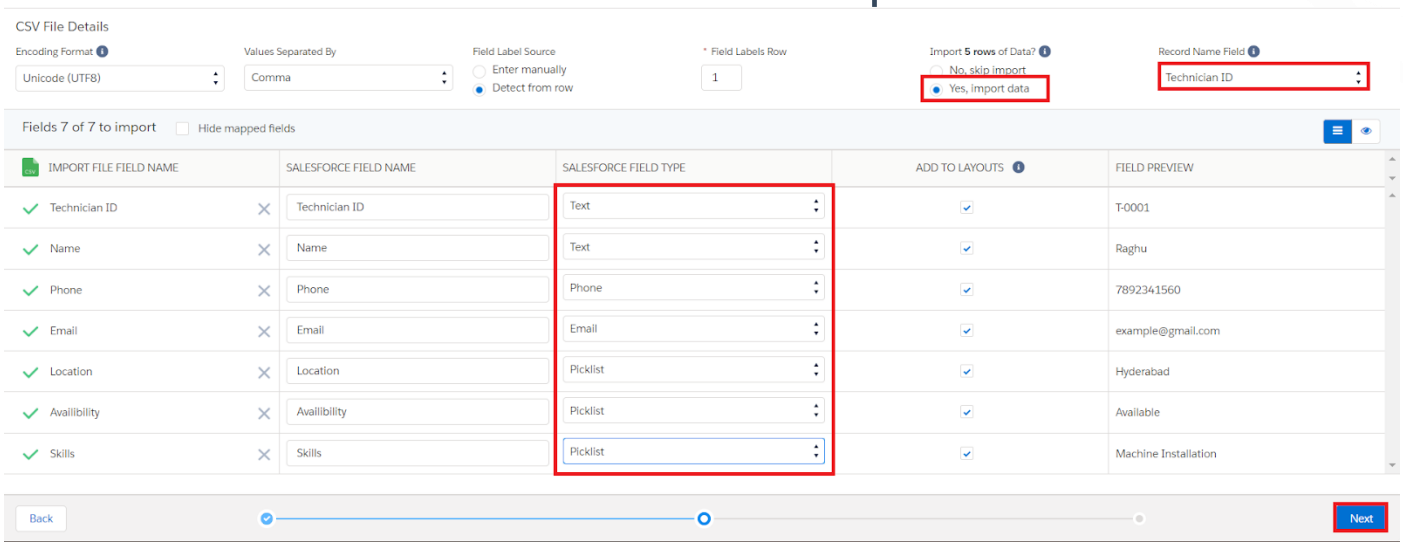
To store the data as per business requirement.

ACTIVITY 1: Create Technician Object

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

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

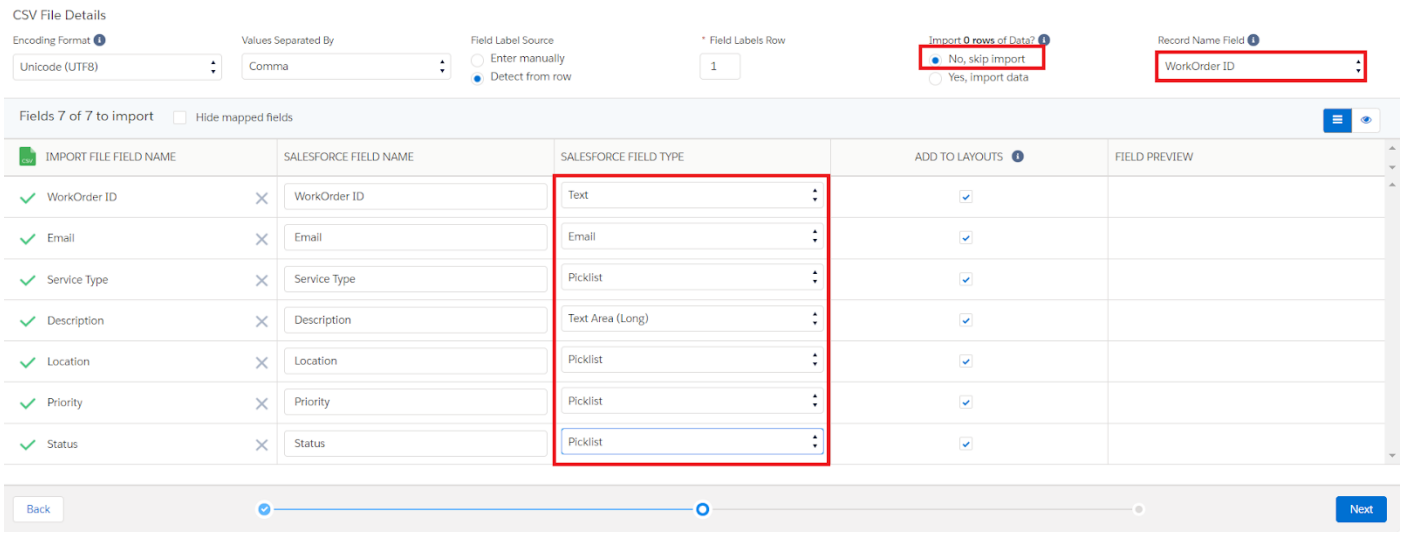
6. Click Login With Salesforce.
7. Enter your Salesforce account username and password. (which you have created in the Milestone 1, Activity 1)
8. Click Log In.
9. Click Allow.
10. Click Upload.
11. Navigate to the Technician.csv file you downloaded and upload it. Salesforce automatically detects the 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.



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

ACTIVITY 2: Create 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.



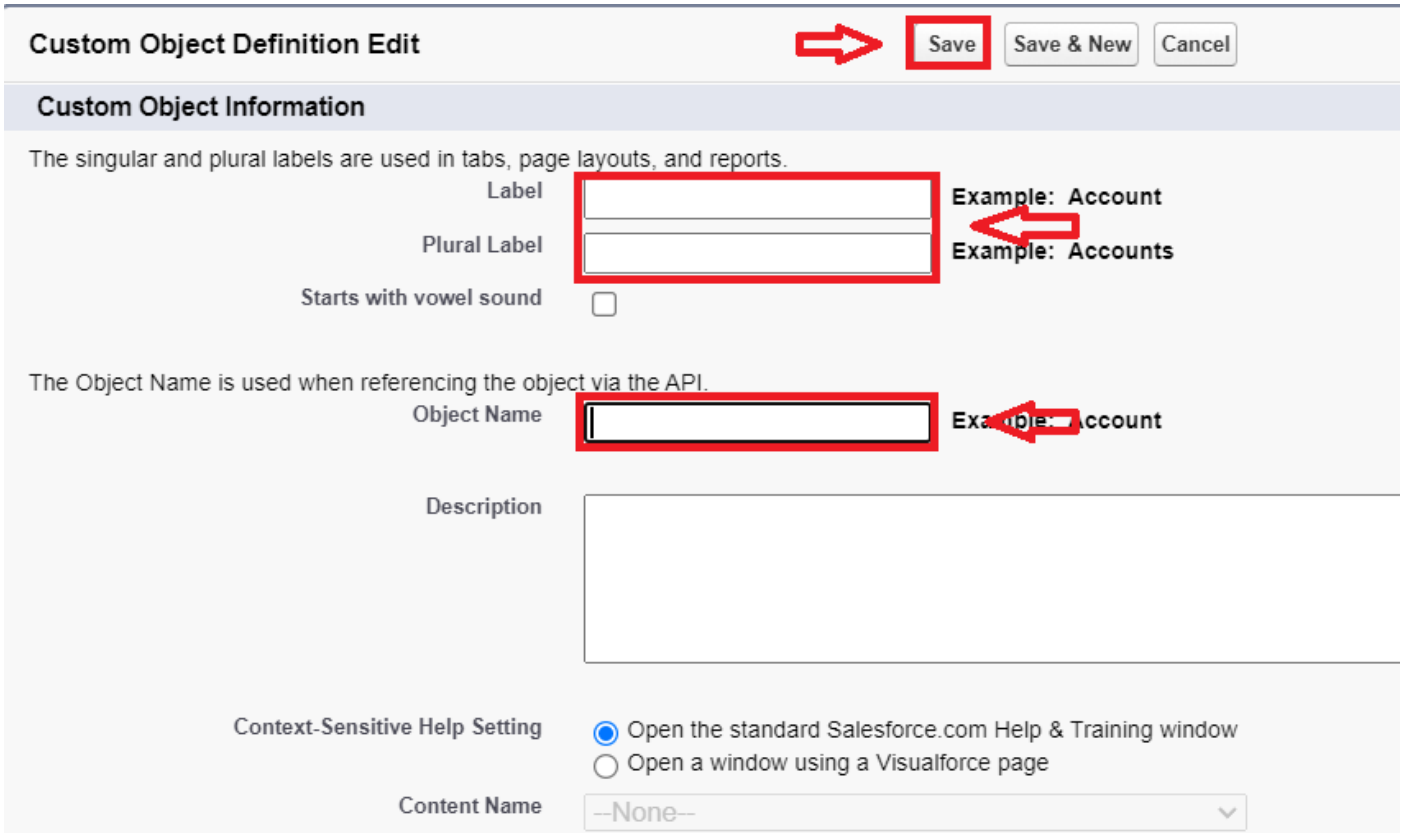
ACTIVITY 3: Create Assignment Object

To create an object:

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

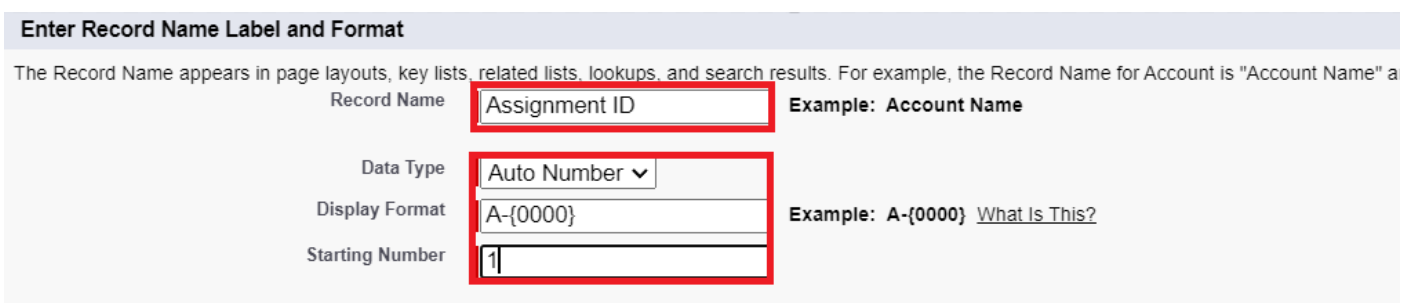
Field service workorder optimization

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



2. Click on Allow reports,

Allow search --> Save.

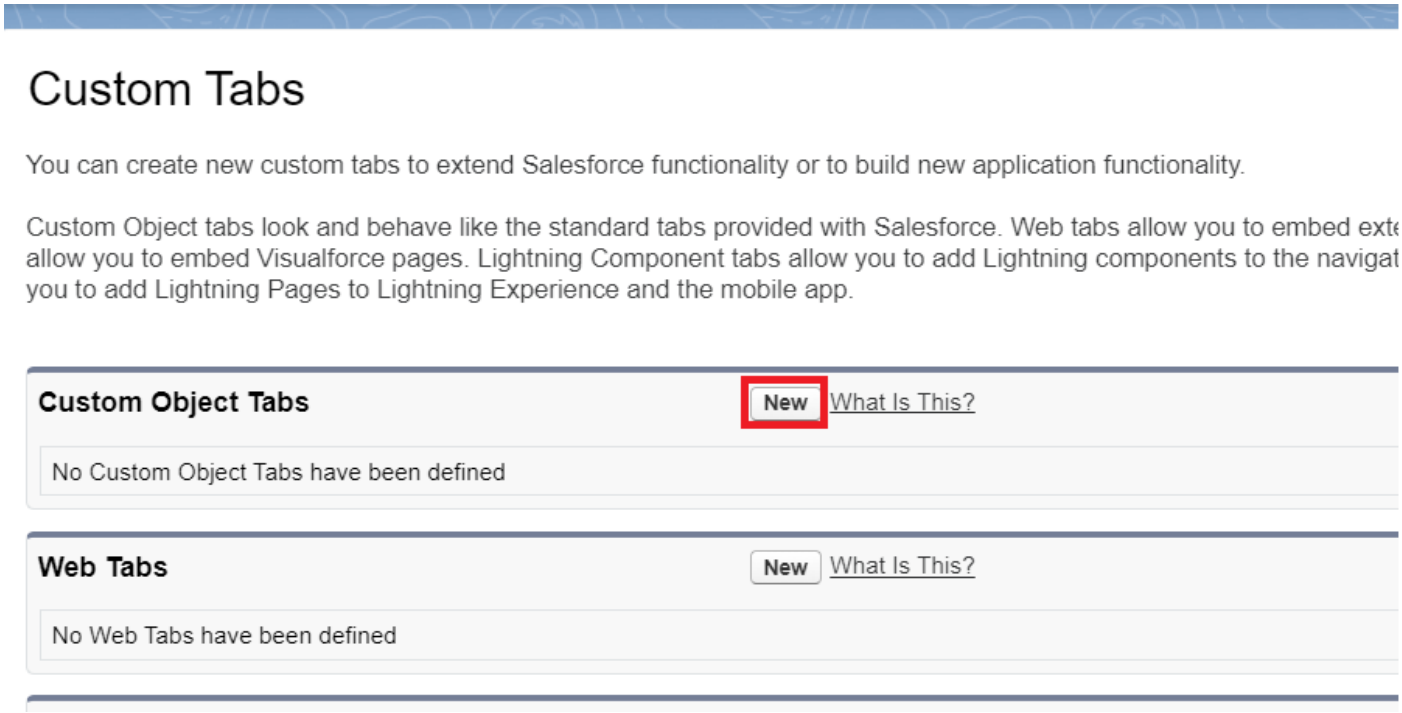
Tabs

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

ACTIVITY 1: Creating a Custom Tab

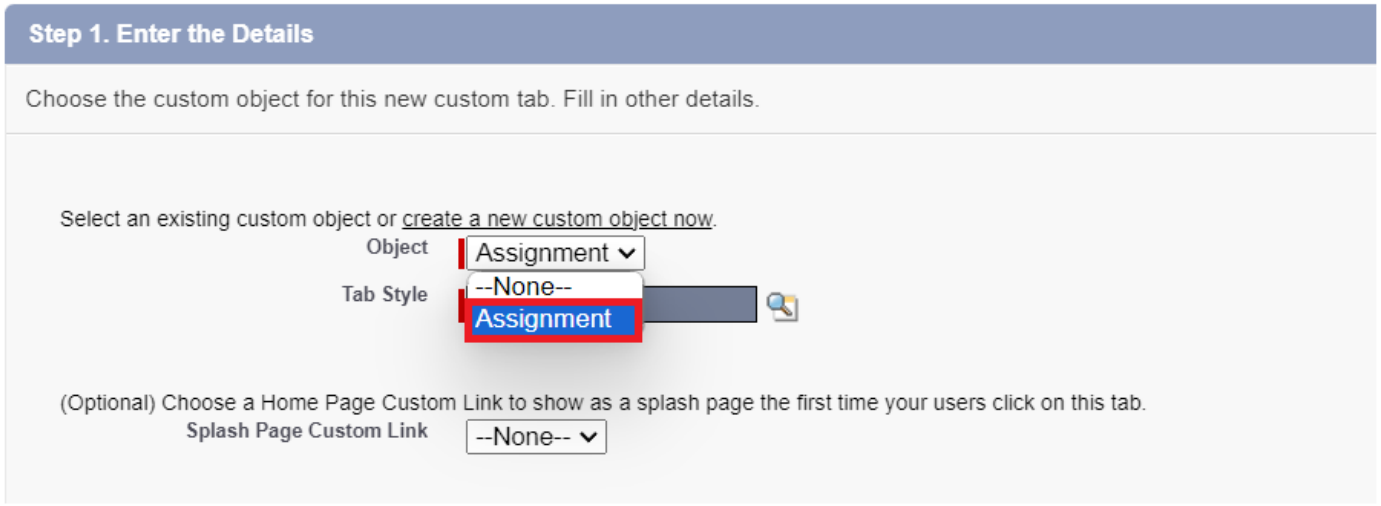
To create a Tab:(Assignment)

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.

New Custom Object Tab



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

The Lightning App

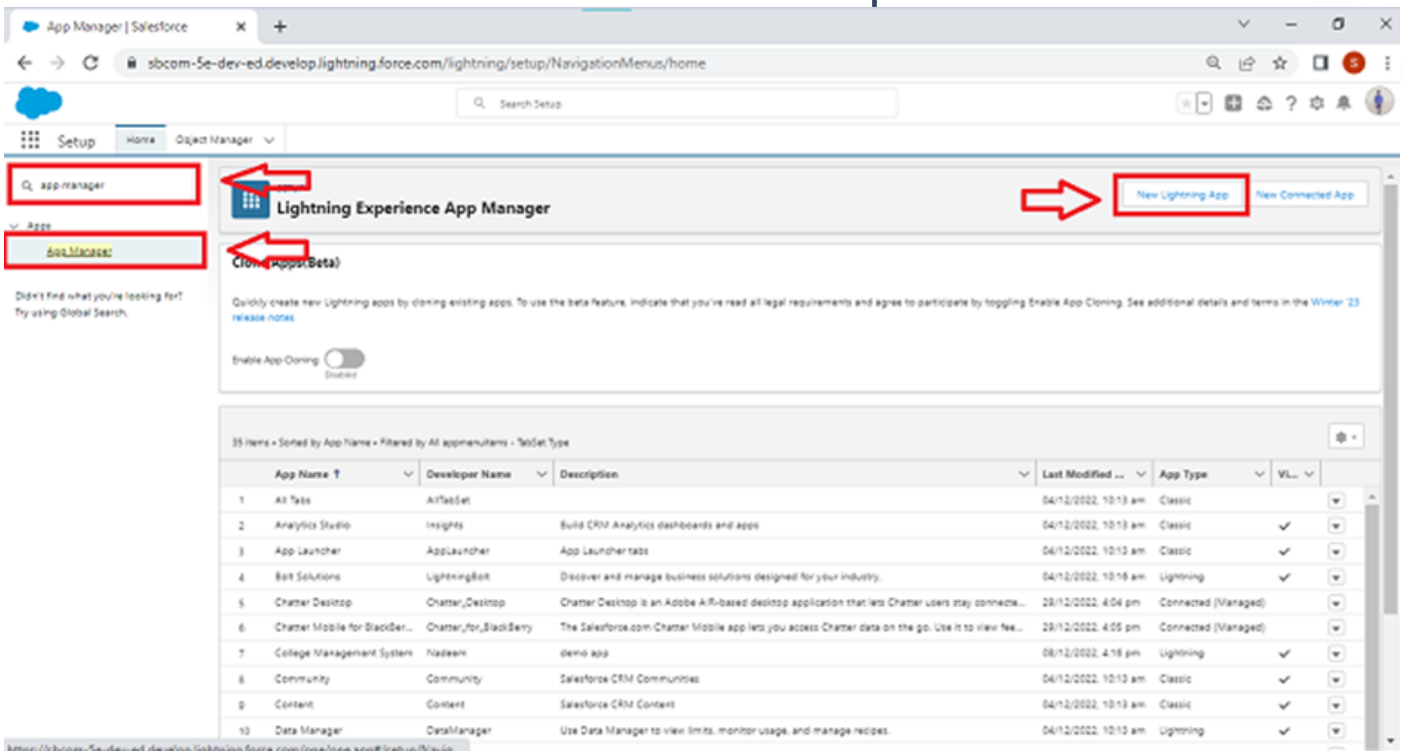
Well done you have reached close to your organizational requirement by creating the objects to store the organization's data. Making a database for an organization is just not enough to reach out the requirements, the task is how the users at the organization can access the objects you have created for them

ACTIVITY 1: Create a Lightning App

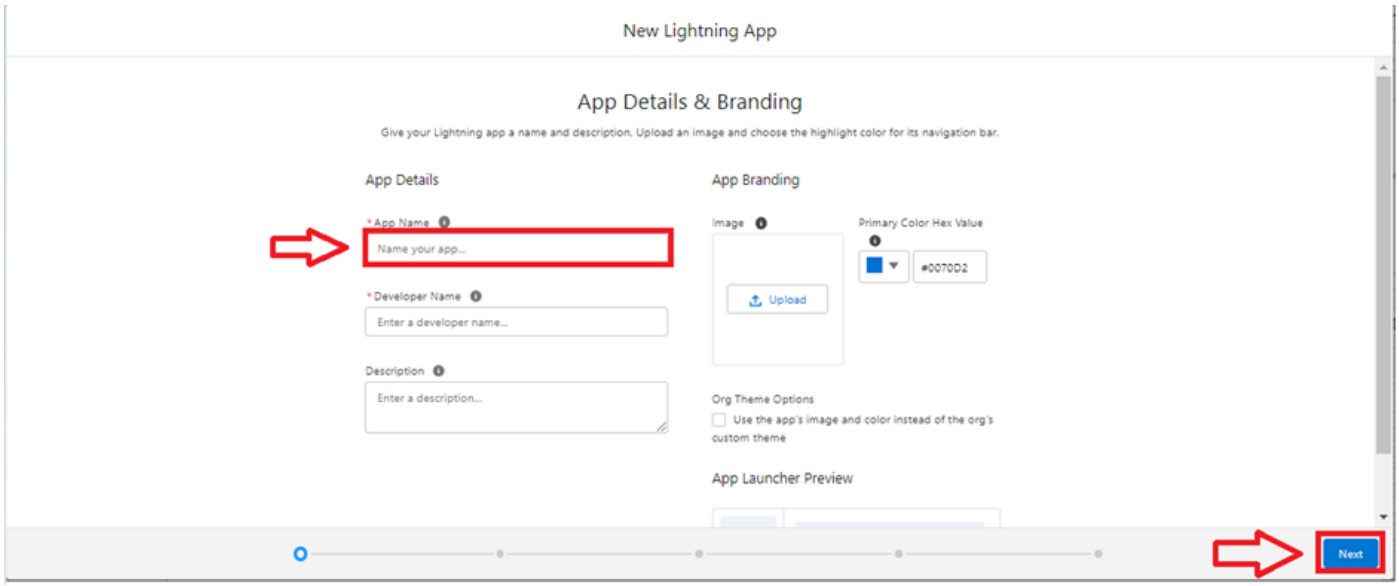
To create a lightning app page:

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

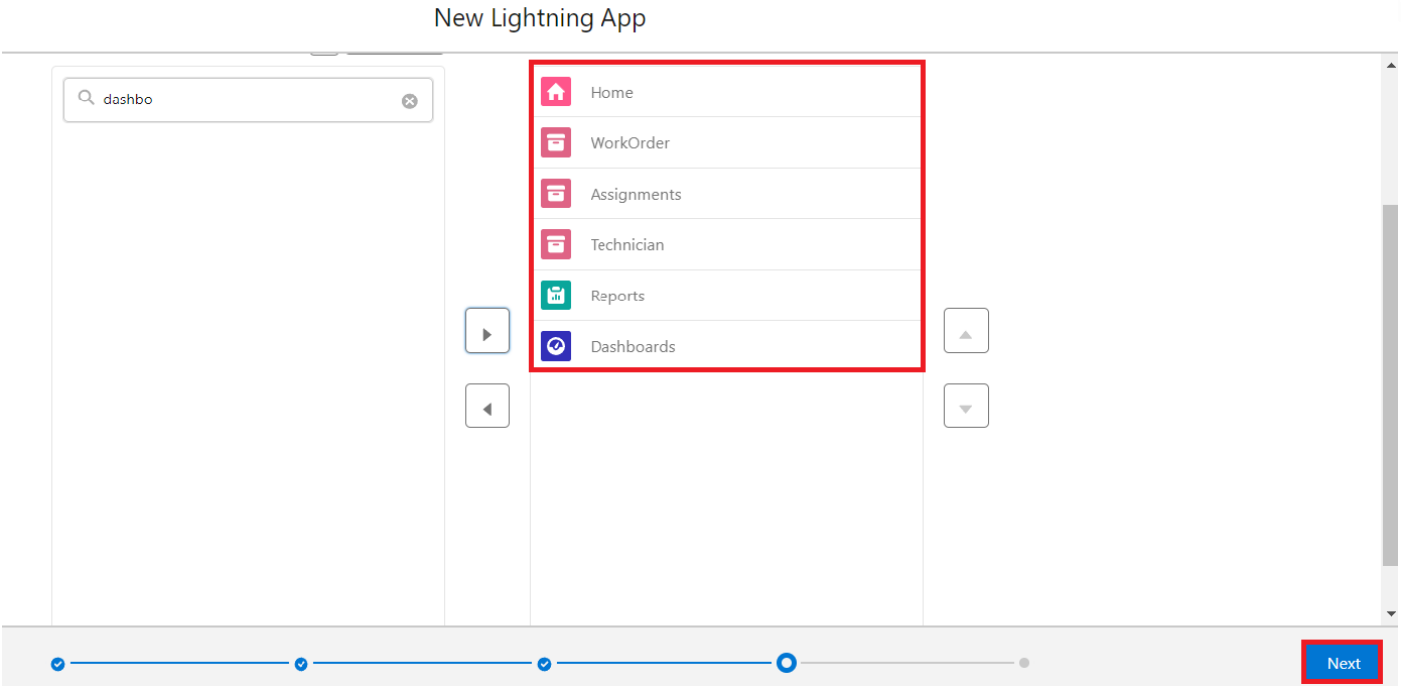
Field service workorder optimization



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.

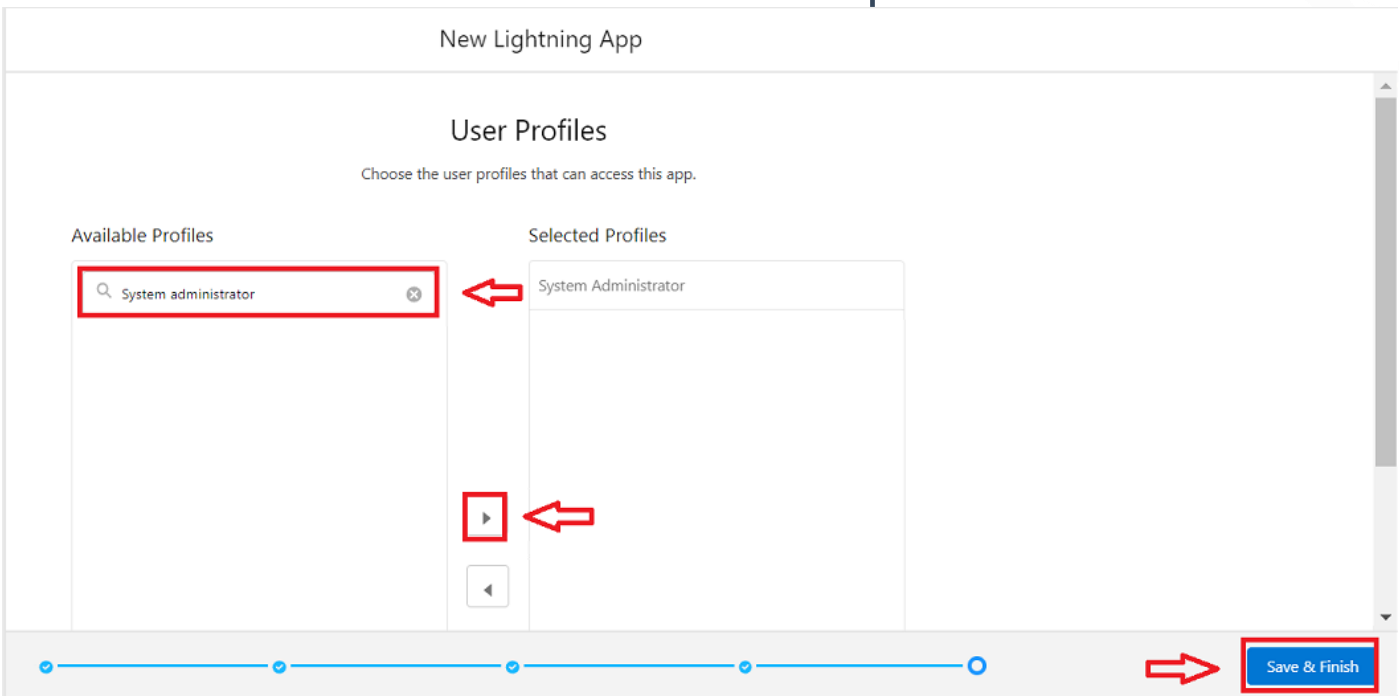


4. To Add Navigation Items:



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.

Fields & Relationship

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

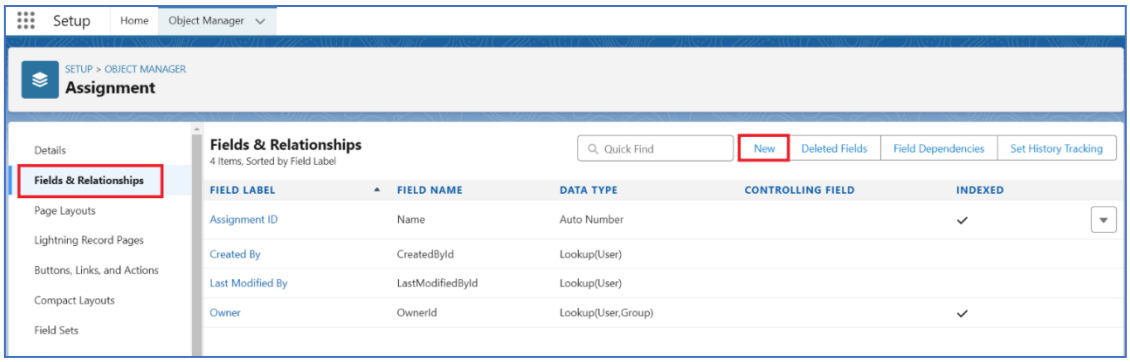
ACTIVITY 1: Creating Lookup Field in Assignment Object

To create fields in an object:

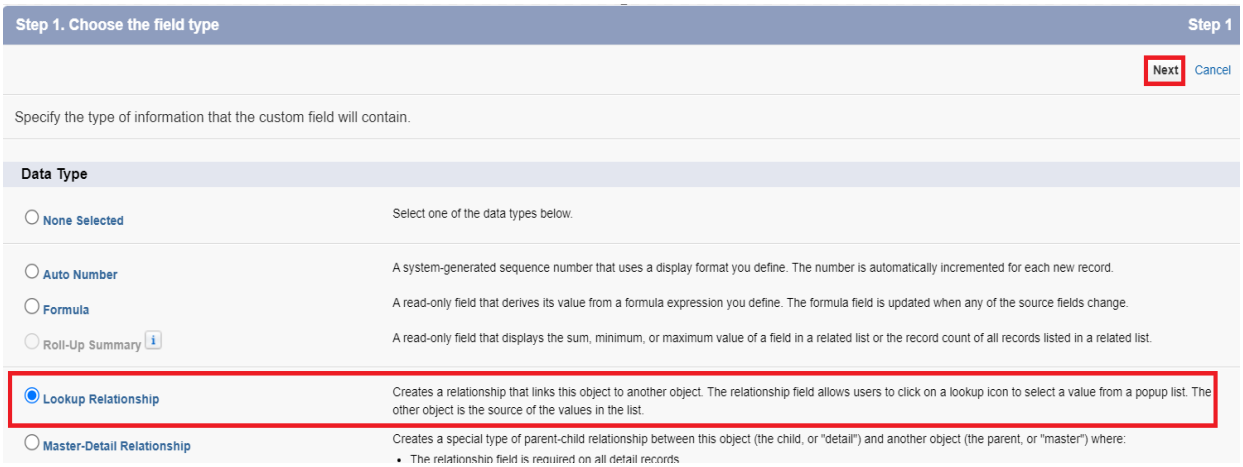
1. Go to setup --> click on Object Manager --> type object name(Assignment) in quick find bar--> click on the object.

2. Now click on "Fields & Relationships" --> New

Field service workorder optimization



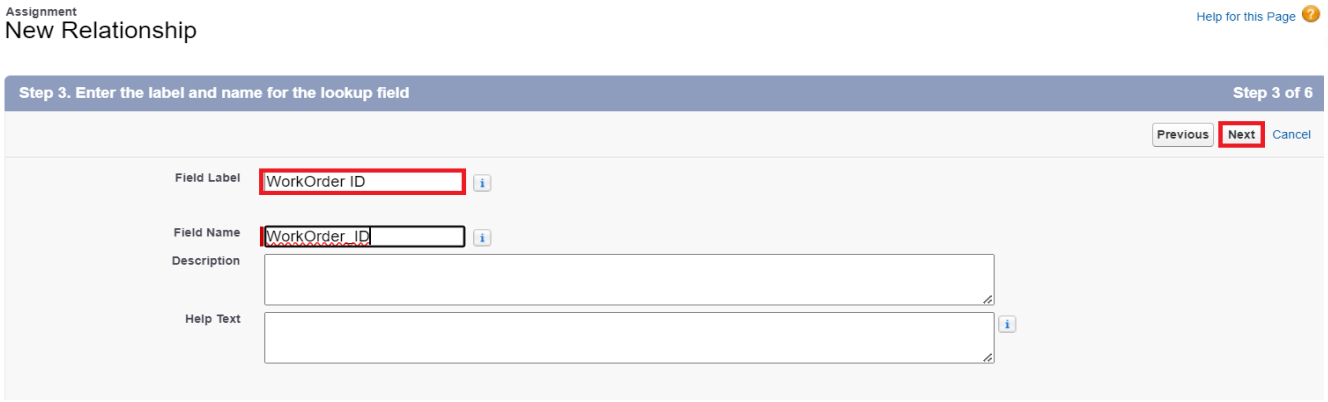
3. Select Data type as “Lookup”.



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.

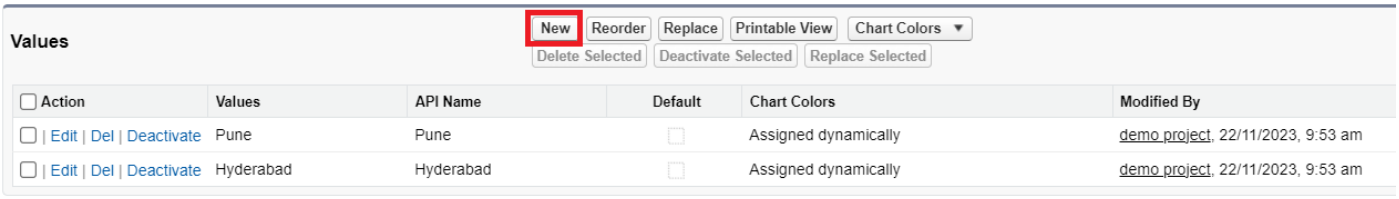
6. Give Field Label as “WorkOrder ID” and click Next.



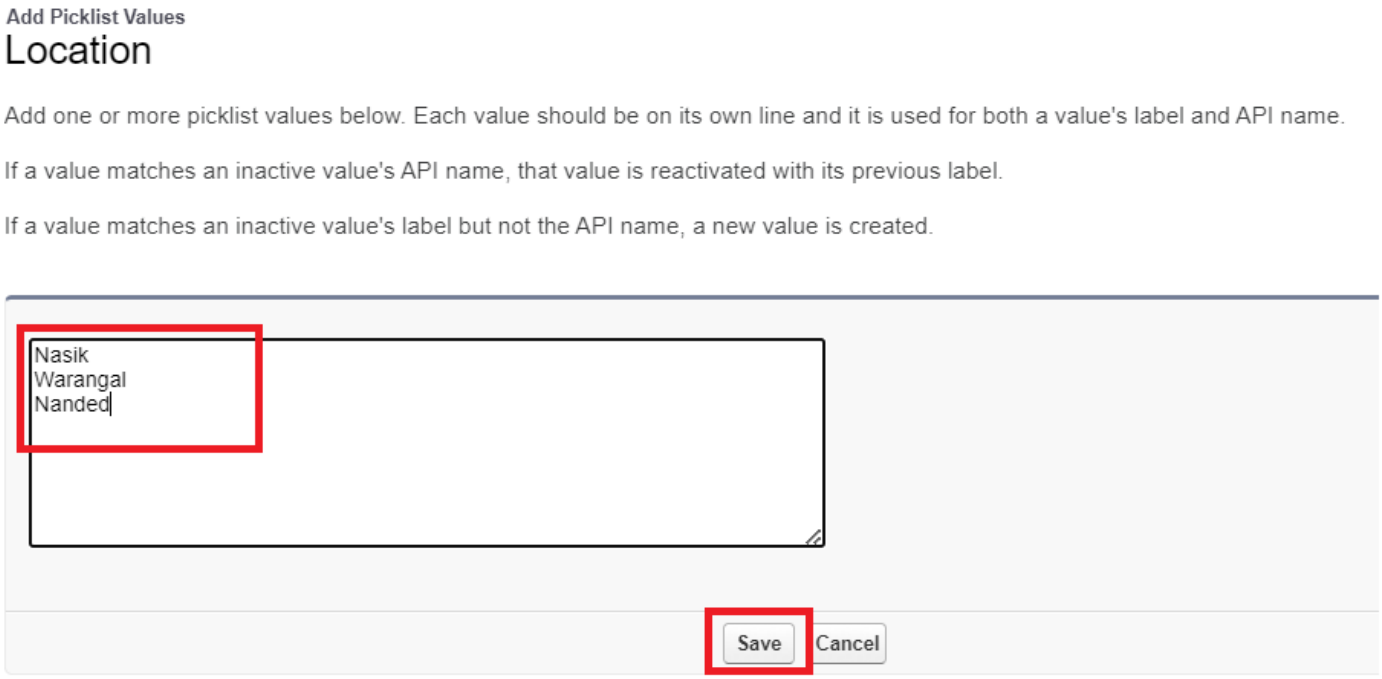
7. Next --> Next --> Save & New.

ACTIVITY 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.



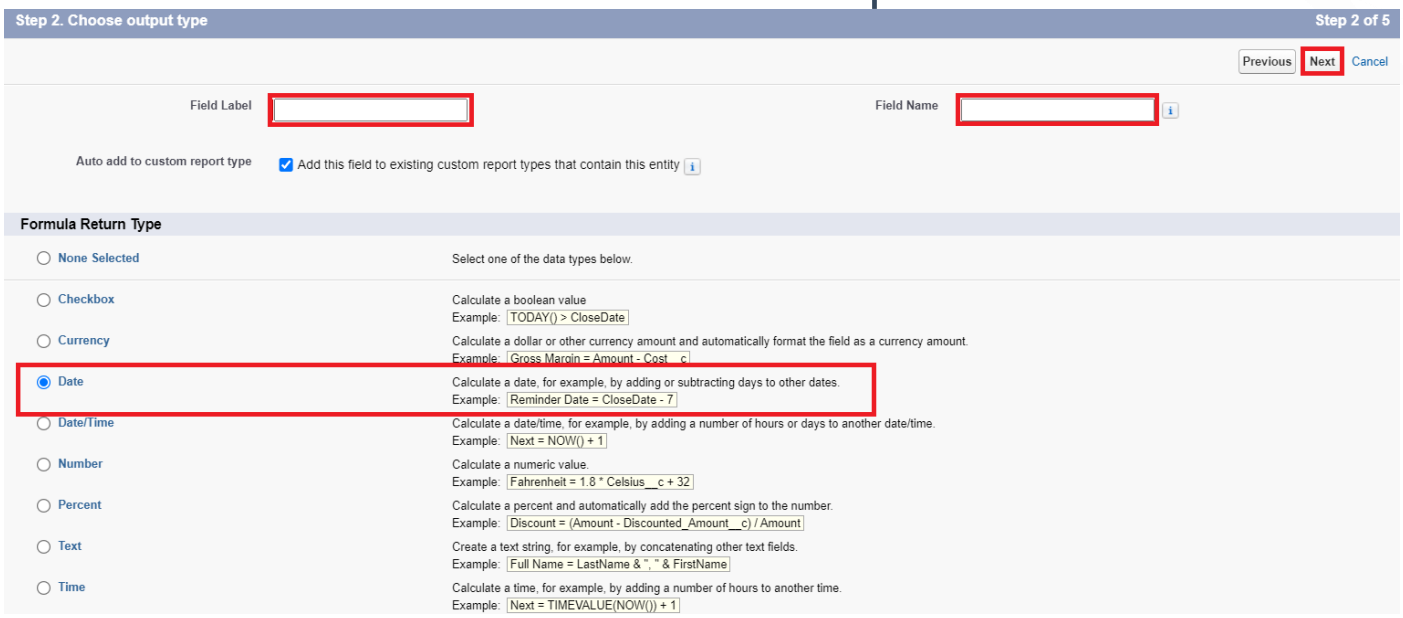
ACTIVITY 3: Manage your picklist values

Add following values to the respective fields in WorkOrder object:

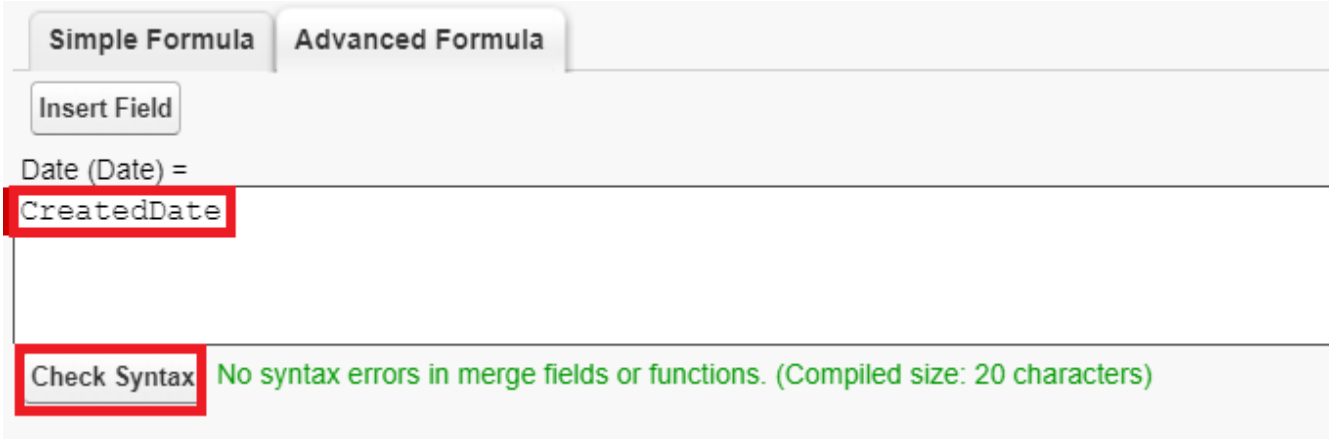
Field	Values
Priority	High
Service Type	Hardware repair Troubleshoot/Debugging Lane-Management

ACTIVITY 4: 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.

Field service workorder optimization

ACTIVITY 5: Creating Remaining fields for the respective objects

Now create the remaining fields using the data types mentioned in the table.

<table>
<tr><th>Sl No</th><th>Object Name</th><th>Field</th></tr>
<tr><td>1</td><td>Assignment</td><td>
<table>
<tr><th>Field Name</th><th>Datatype</th></tr>
<tr><td>• Technician ID
• Assignment Date

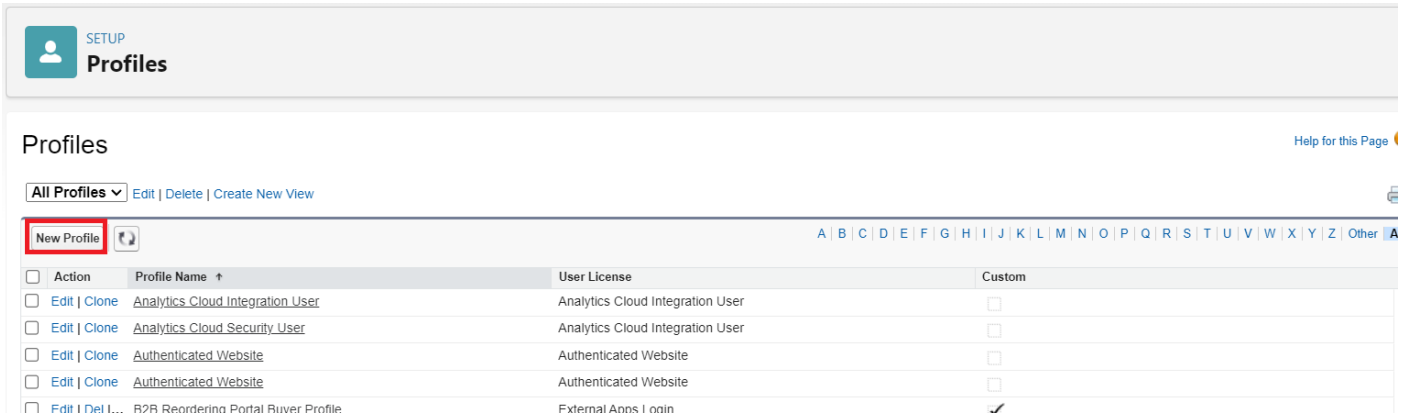
• Completion Date</td><td>Lookup(Technician)
Formula: return type : Date
(WorkOrder_ID__r.Date__c)
Formula: return type : Date
IF(ISPICKVAL(WorkOrder_ID__r.Status__c , 'Resolved'), WorkOrder_ID__r.LastModifiedDate , NULL)</td></tr>
</table>
</td></tr>
</table>

Profiles

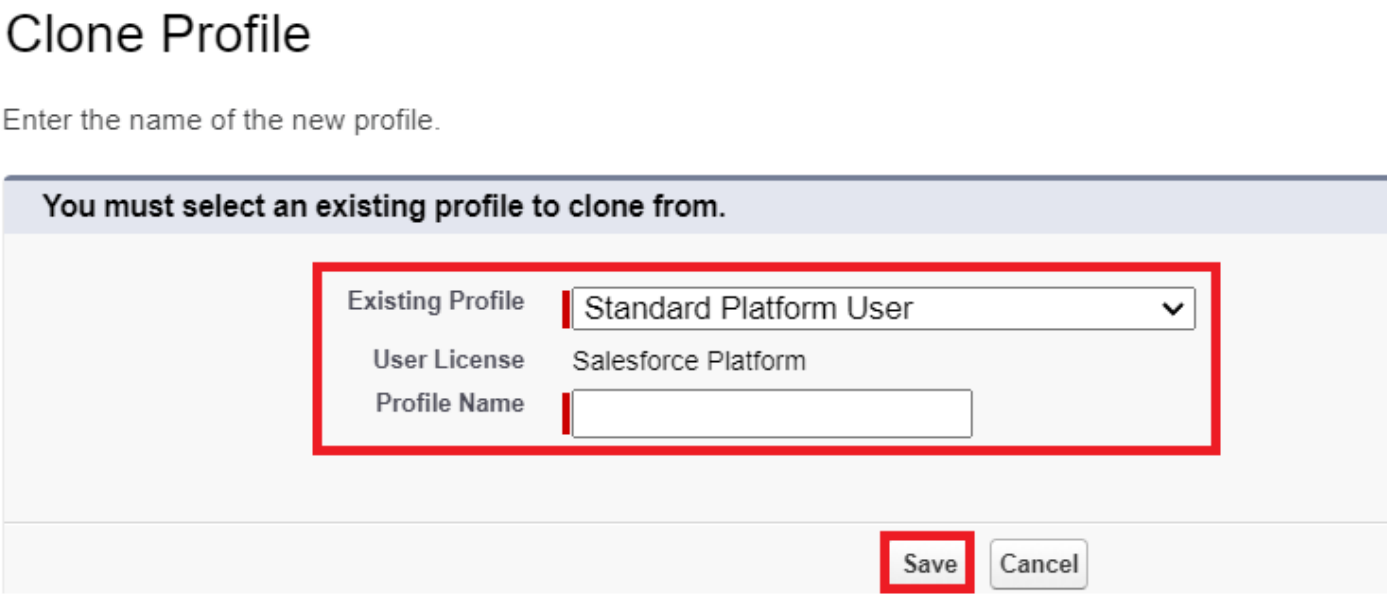
Profile defines what an user is able to do or see in the Salesforce Org

ACTIVITY 1: Technician Profile

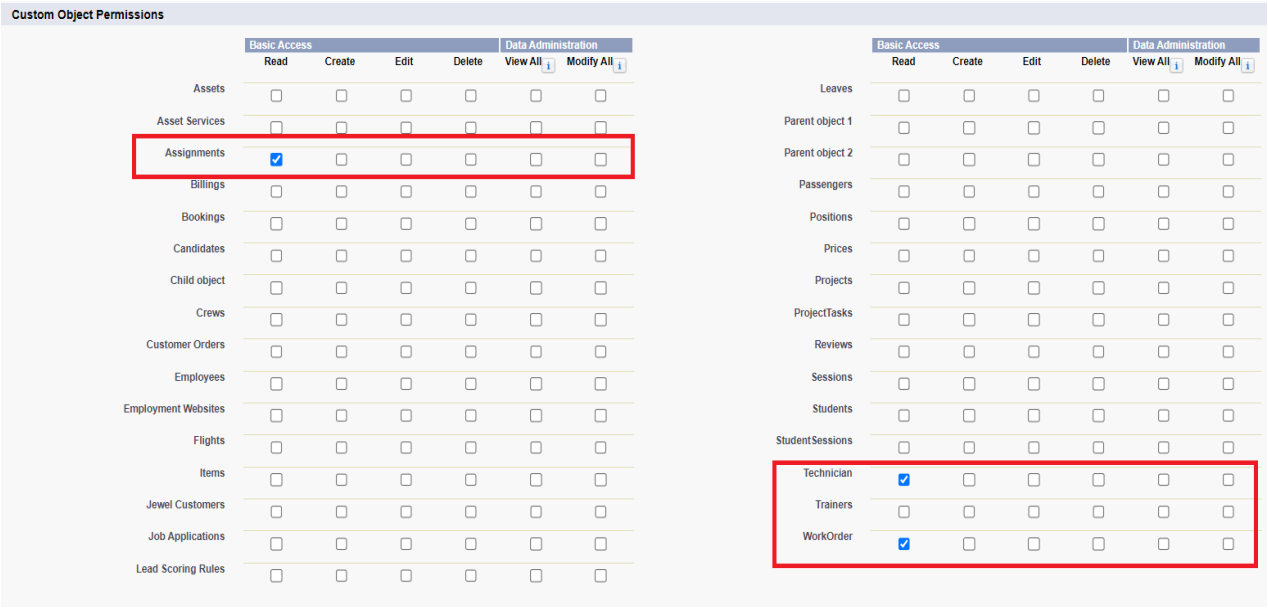
1. Go to setup --> type profiles in quick find box --> click on profiles --> click on new profile.



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:



Field service workorder optimization

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 Save.

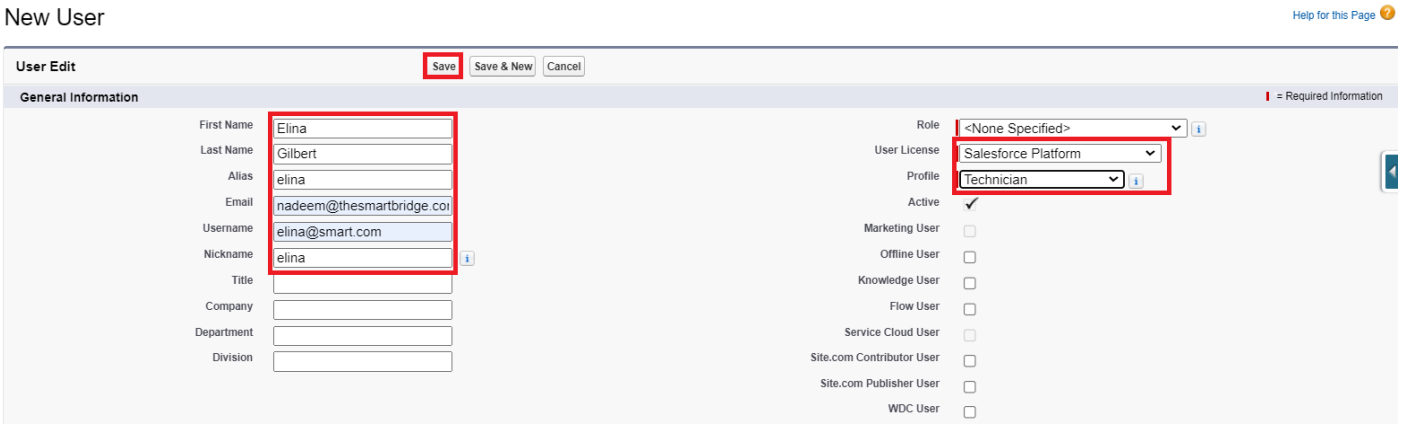
Field service workorder optimization

Users

Users are defined as the employees of your organization

ACTIVITY 1: 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

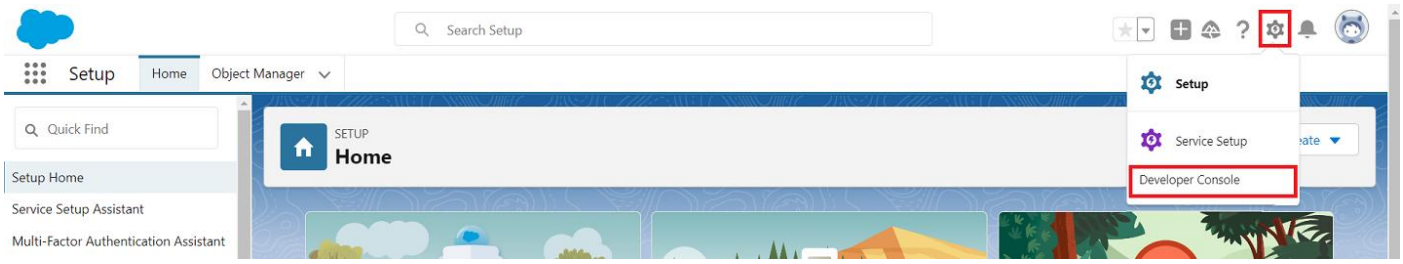


3. Save

Apex Trigger

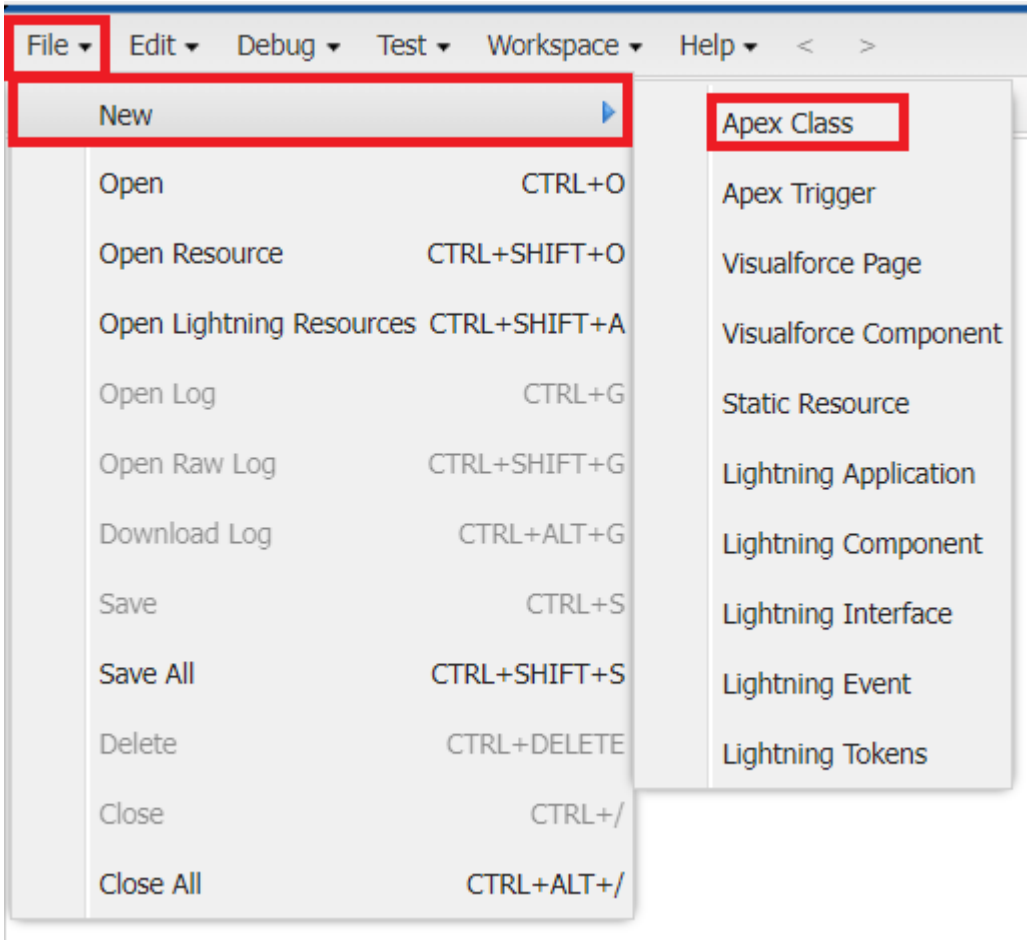
ACTIVITY 1: Create an Apex Class

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

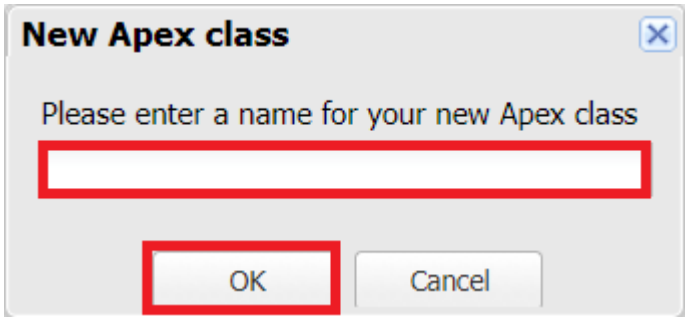


2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.

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

```
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){
```

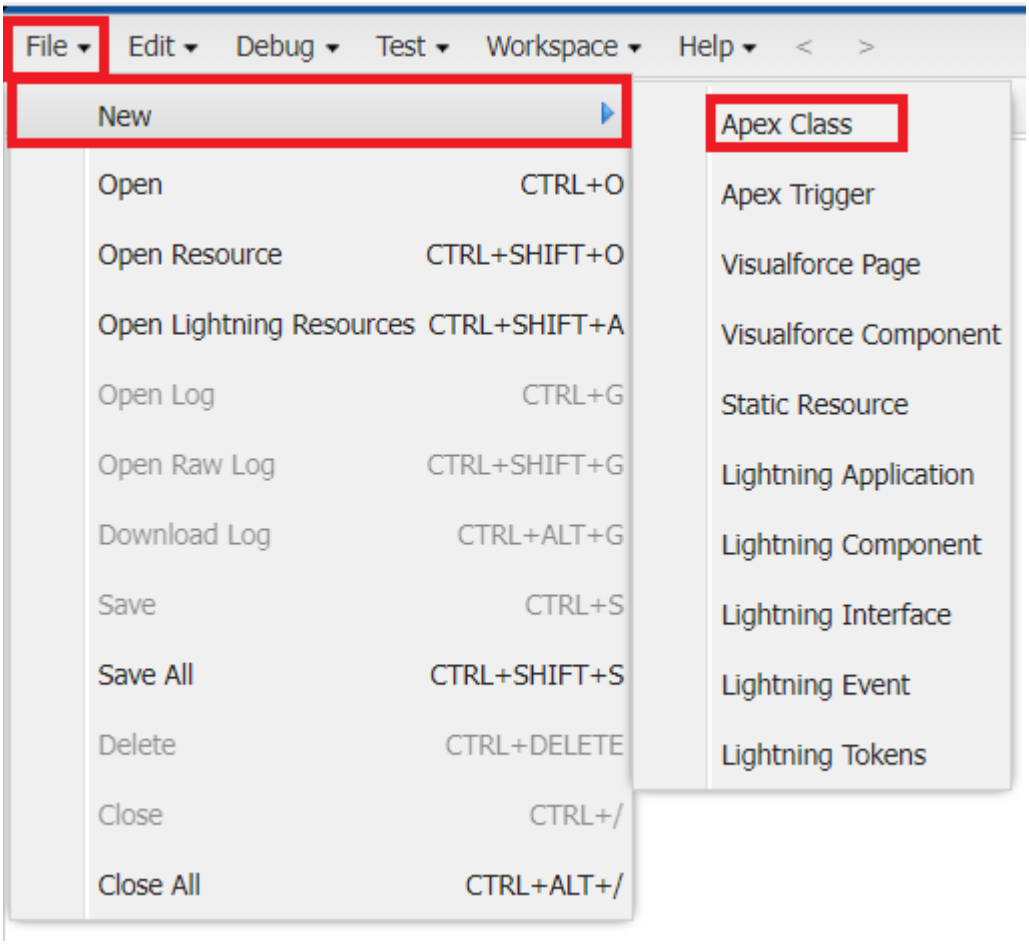
Source Code:

```
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;
        }
    }
}
```

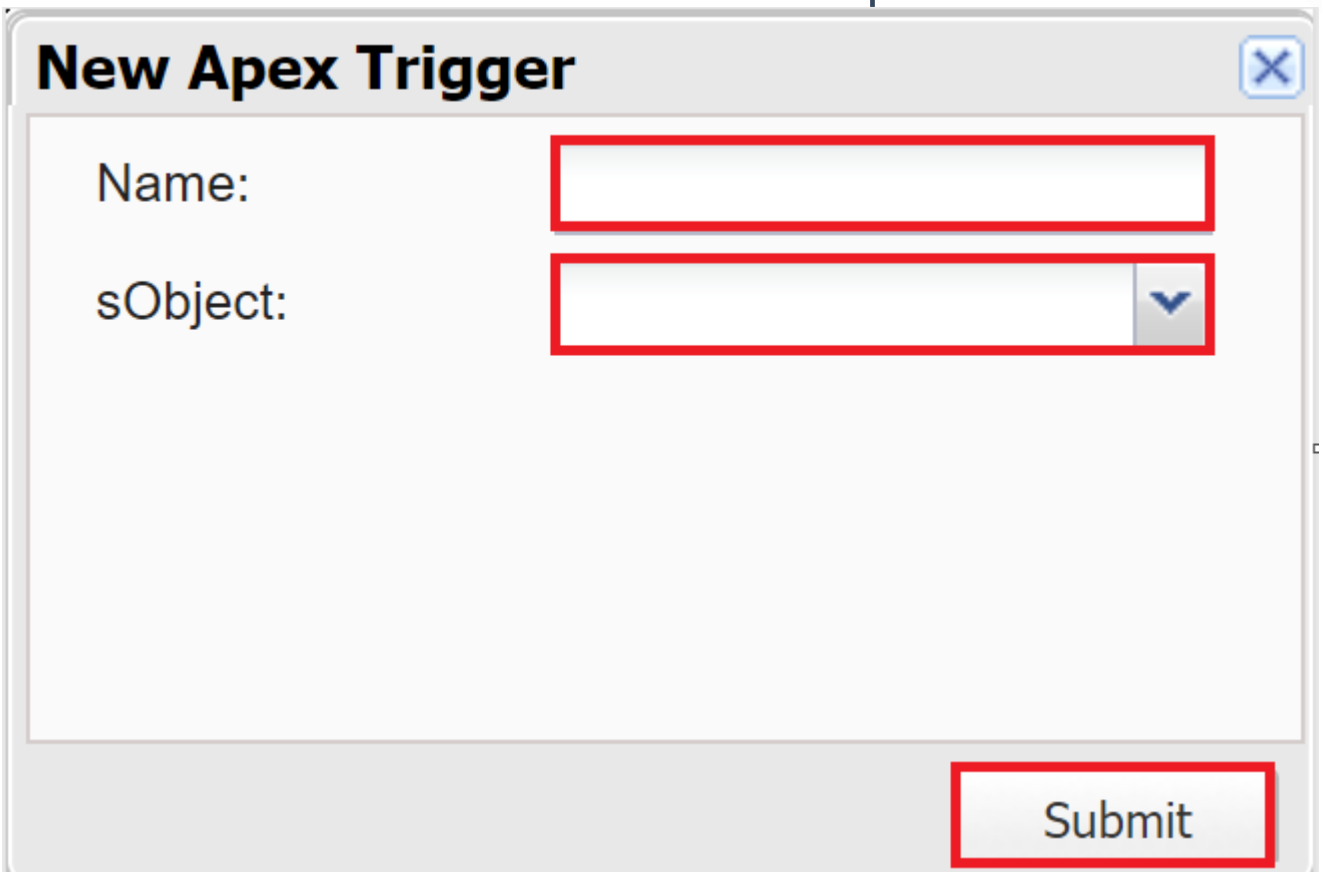
7. Save the code.(click on file --> Save)

ACTIVITY 2 : Create an Apex Trigger

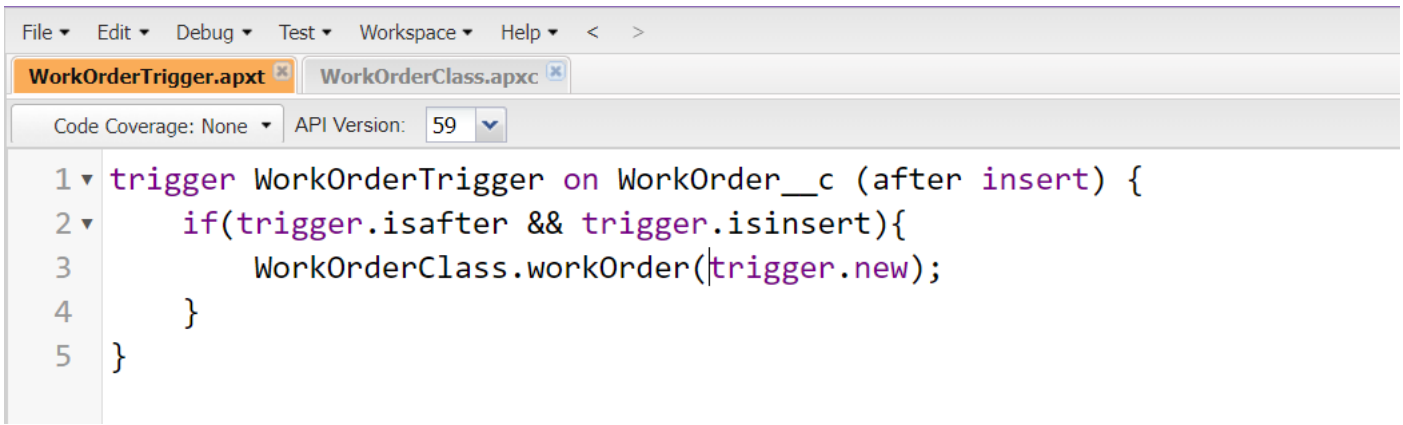
1. To create a new Apex Class follow the below steps:
Click on the file --> New --> Apex Class.



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:

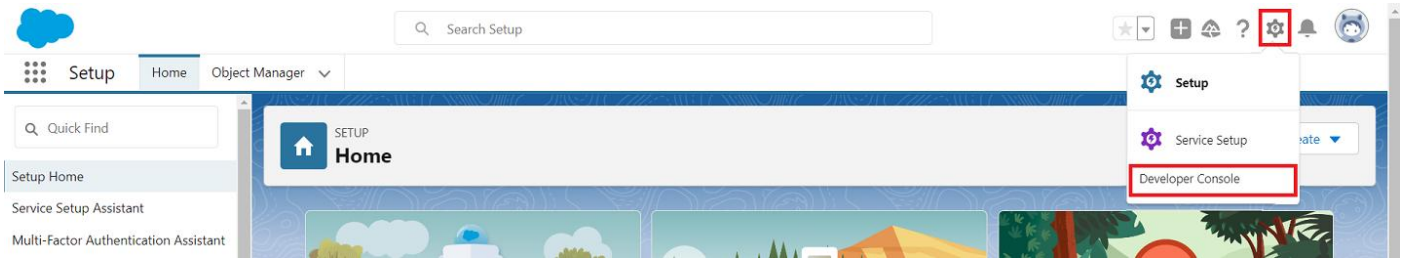
```
trigger WorkOrderTrigger on WorkOrder__c (after insert) {
    if(trigger.isafter && trigger.isinsert){
        WorkOrderClass.workOrder(trigger.new);
    }
}
```

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

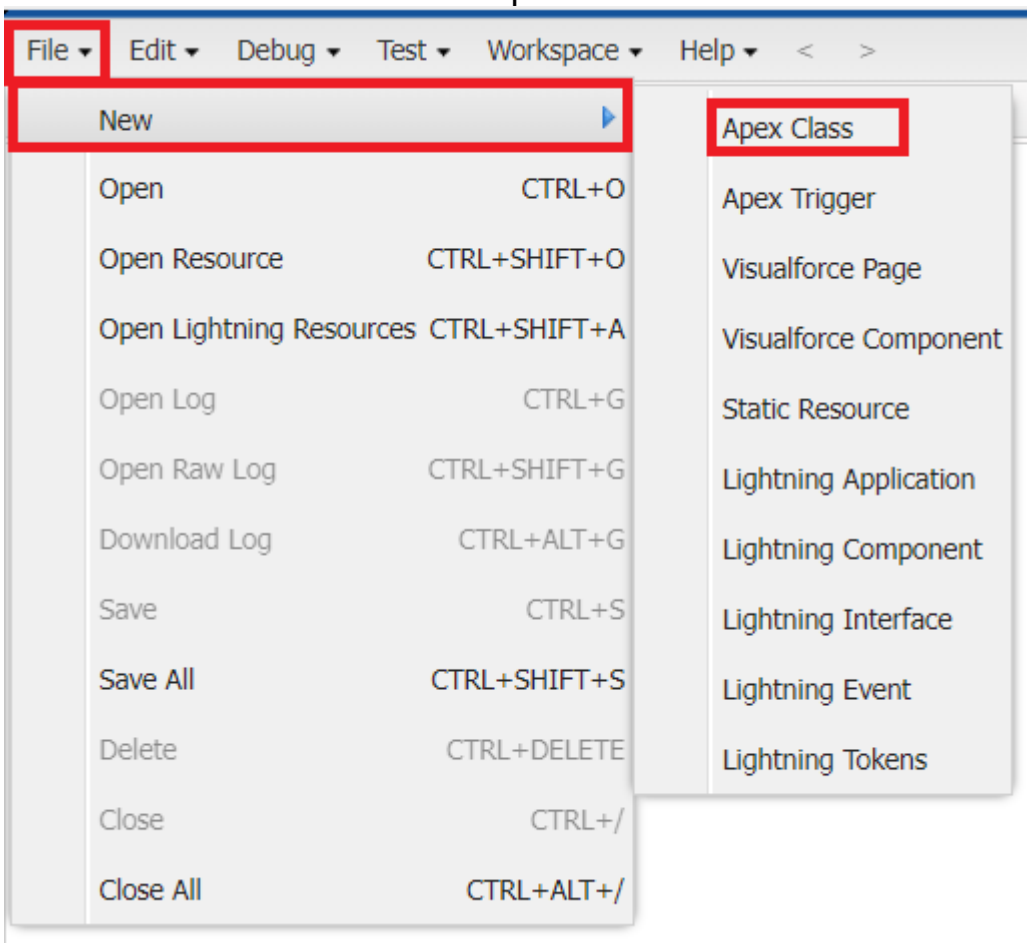
Field service workorder optimization

ACTIVITY 3 : Create an Apex Class

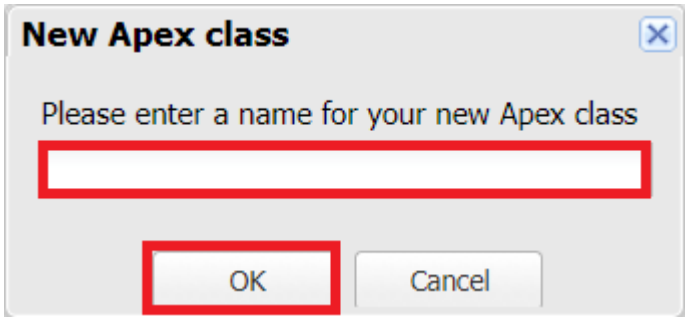
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



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”.



4. Click ok.
5. 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());
        }
    }
}
```

Source Code:

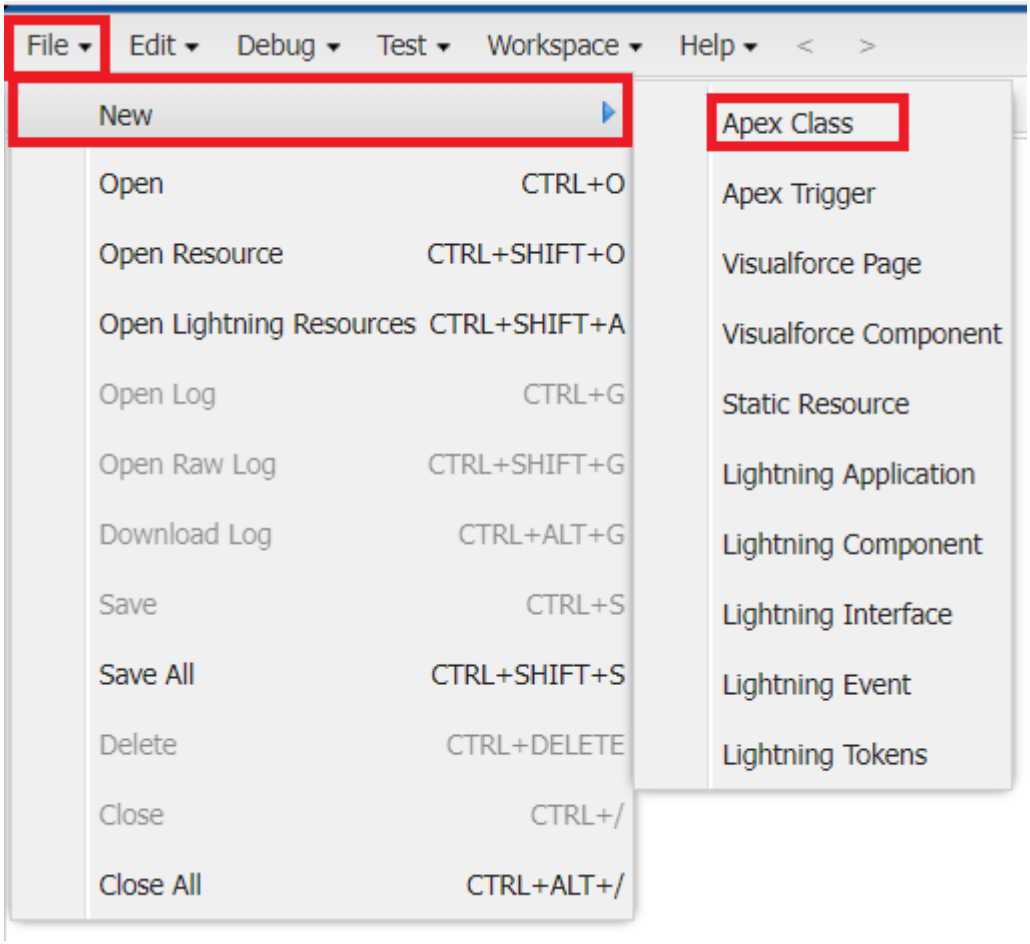
```
public class AssigningEmail {
    public static void sendEmailmsg(List<Assignment__c> assRec){
        List<messaging.SingleEmailMessage> myVar = new List<messaging.SingleEmailMessage>();
        Map<id,Technician__c> tecnicians = new Map<id,Technician__c>([SELECT Id, Phone__c, Location__c,
Skills__c,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

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

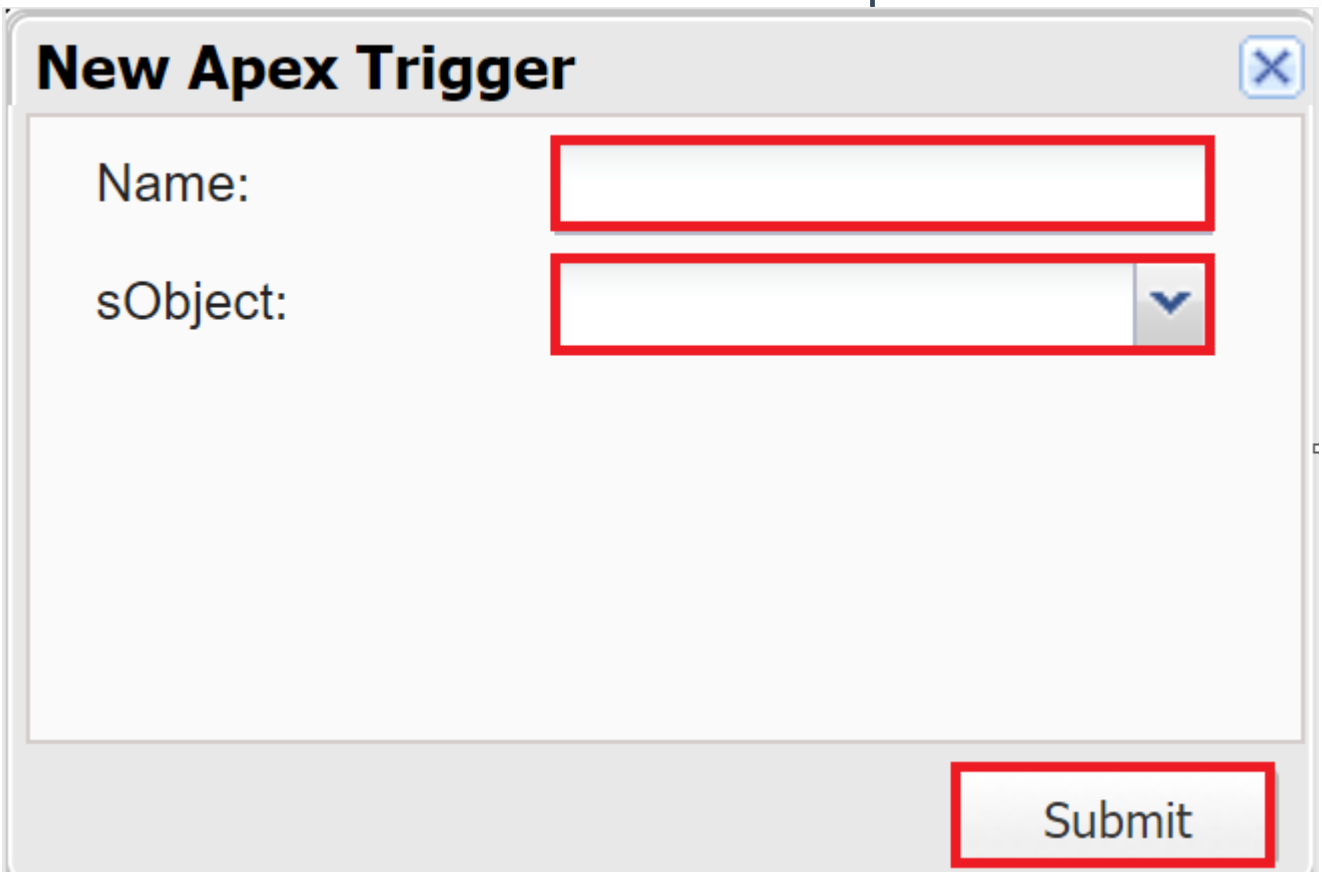
ACTIVITY 4: Create an Apex Trigger

To create a new Apex Class follow the below steps:

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

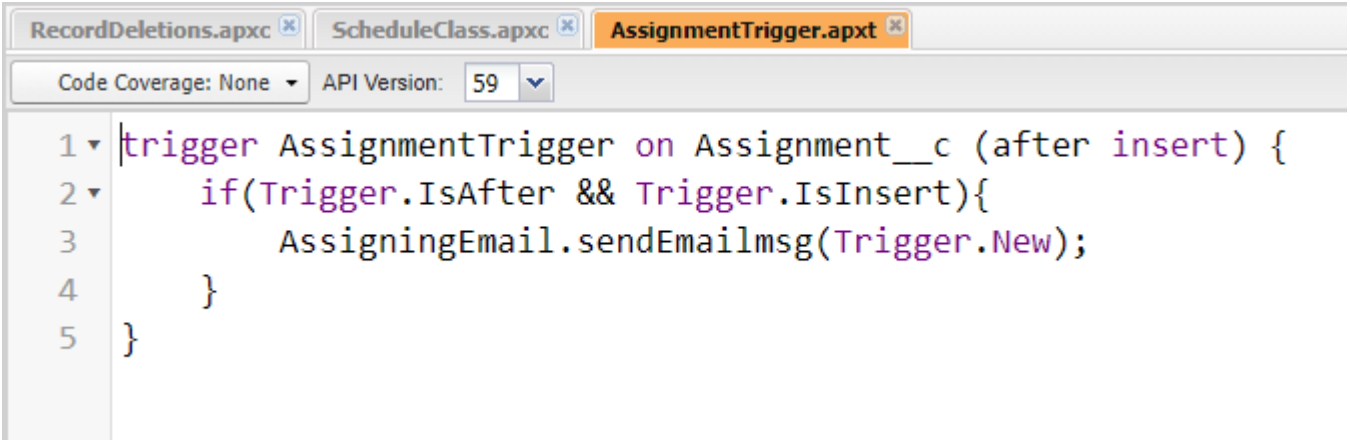


2. Give the Apex Trigger name as “AssignmentTrigger”, and select “Assignment__c” from the dropdown for sObject.



3. Click Submit.

4. Now write the code logic here



Source Code:

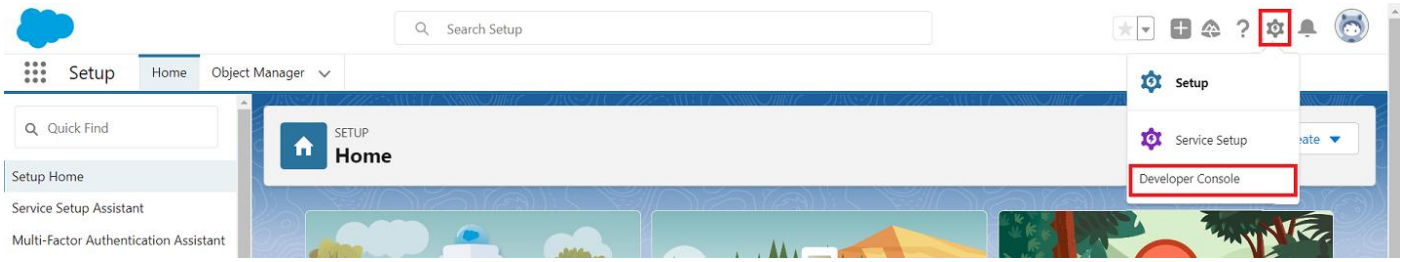
```
trigger AssignmentTrigger on Assignment__c (after insert) {
    if(Trigger.IsAfter && Trigger.IsInsert){
        AssigningEmail.sendEmailmsg(Trigger.New);
    }
}
```

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

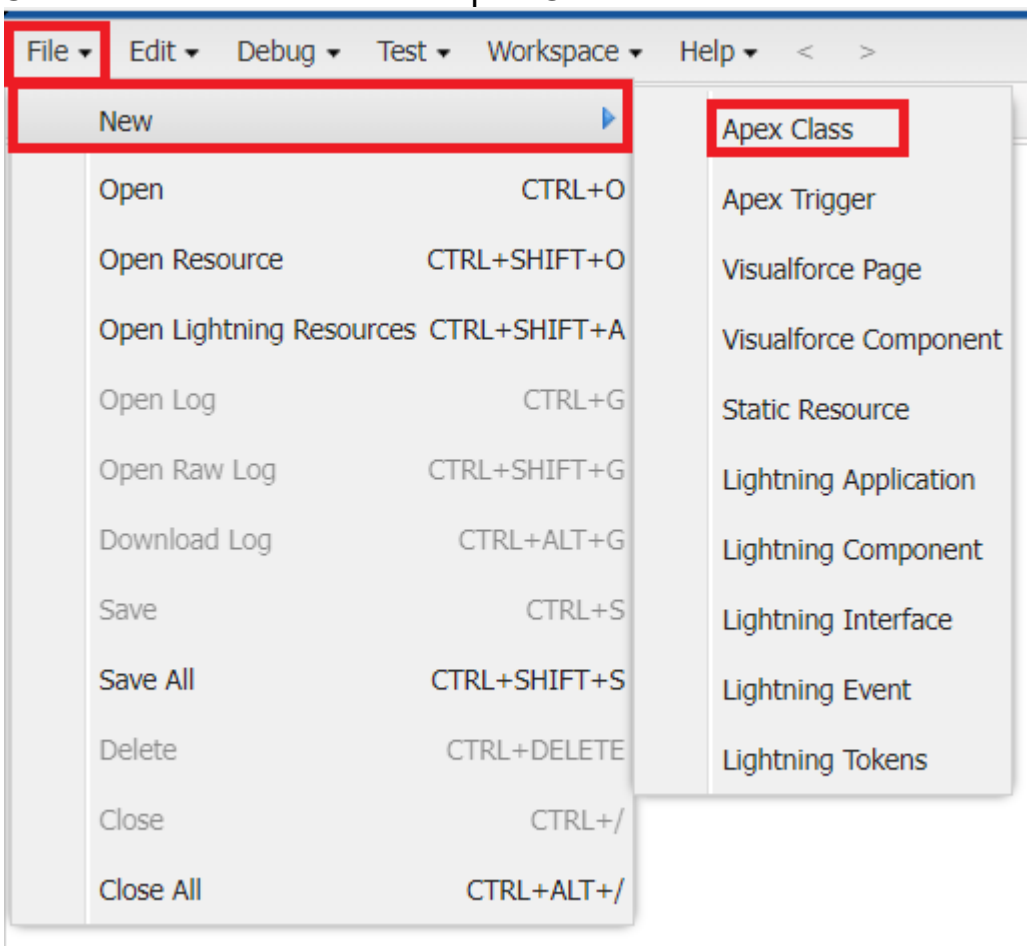
Field service workorder optimization

ACTIVITY 5 : Create an Apex Class

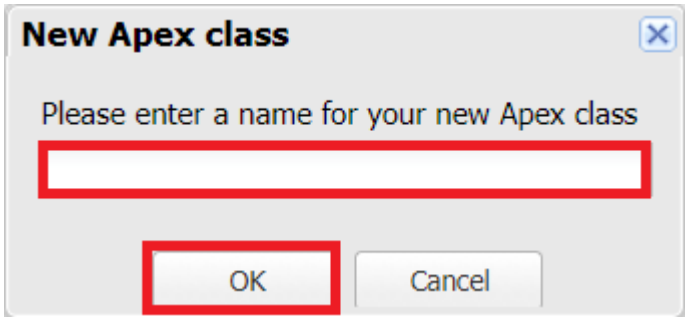
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



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

RecordDeletions.apxc | ScheduleClass.apxc | CompletionMail.apxc

Code Coverage: None | API Version: 59

```
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);
    }
}
```

Source Code:

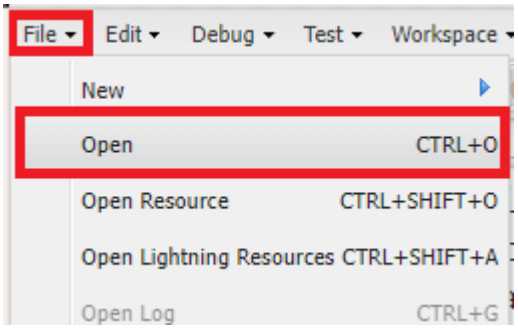
```
public class CompletionMail {
    public static void sendEmailMsg(List<WorkOrder__c> workOrderList){
        List<messaging.SingleEmailMessage> myVar = new
List<messaging.SingleEmailMessage>();
        for(WorkOrder__c con : workOrderList){
            if(con.Status__c == 'Resolved'){
                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. Save the code.(click on file --> Save)

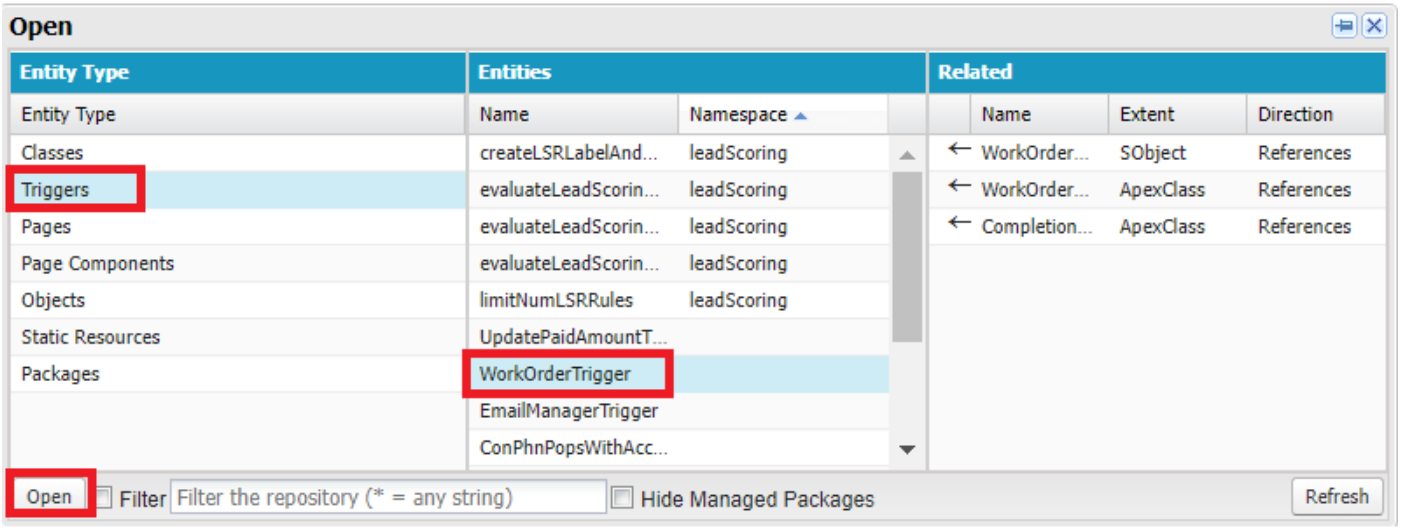
Field service workorder optimization

ACTIVITY 6 : Create an Apex Trigger

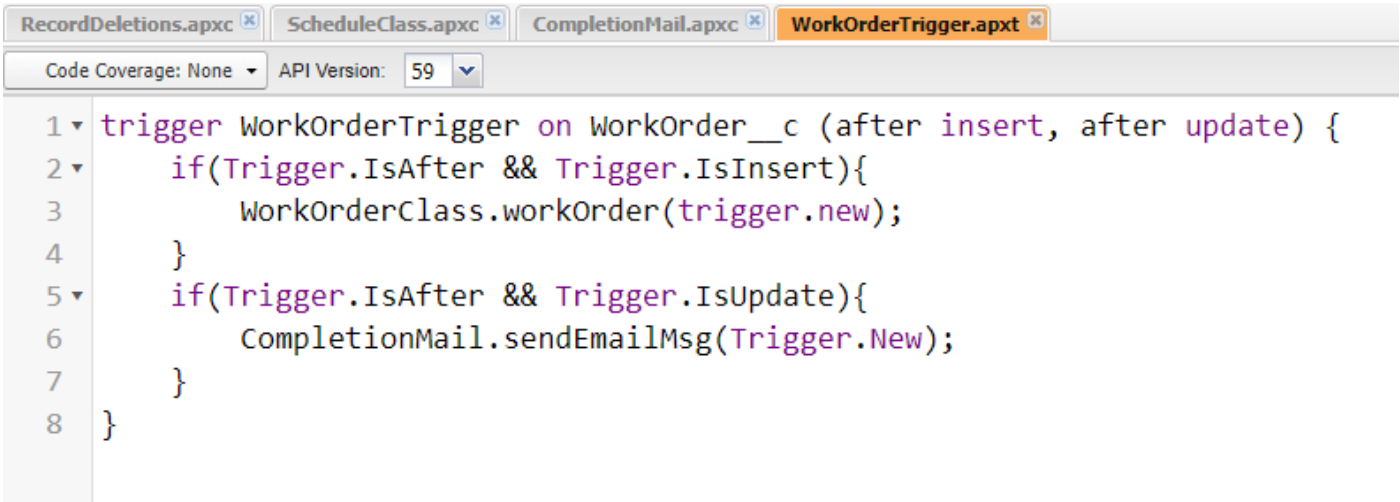
1.Click on the file --> Open.



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



3. Now write the code logic here.



```
 WorkOrderClass.workOrder(trigger.new);
    }
    if(Trigger.IsAfter && Trigger.IsUpdate){
        CompletionMail.sendEmailMsg(Trigger.New);
    }
}
```

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

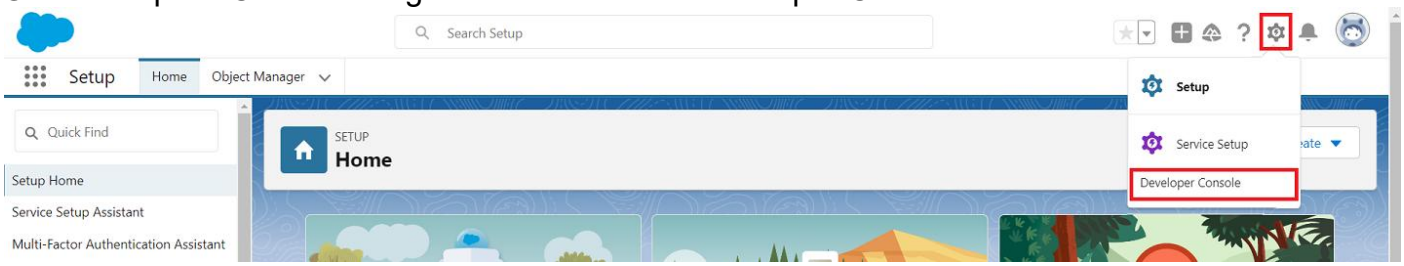
ACTIVITY 7 : Create an Asynchronous Apex Class

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

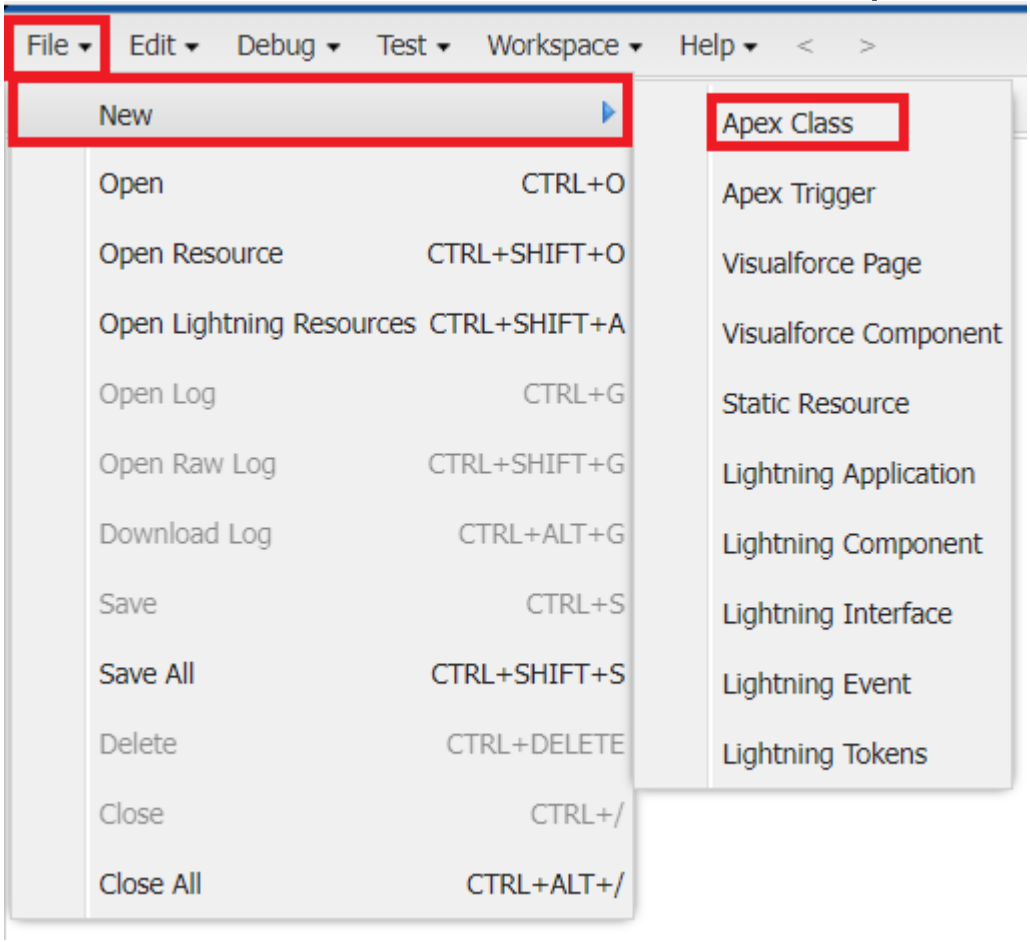
1. Go to Setup --> Click on the gear icon --> Select Developer Console.



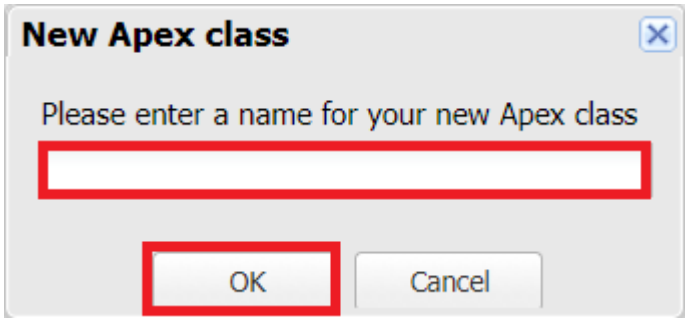
2. Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.

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

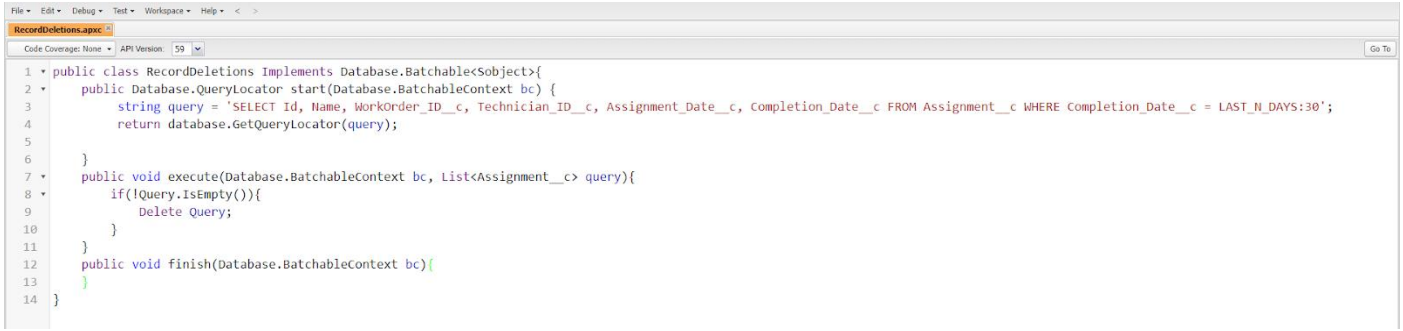
Field service workorder optimization



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



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



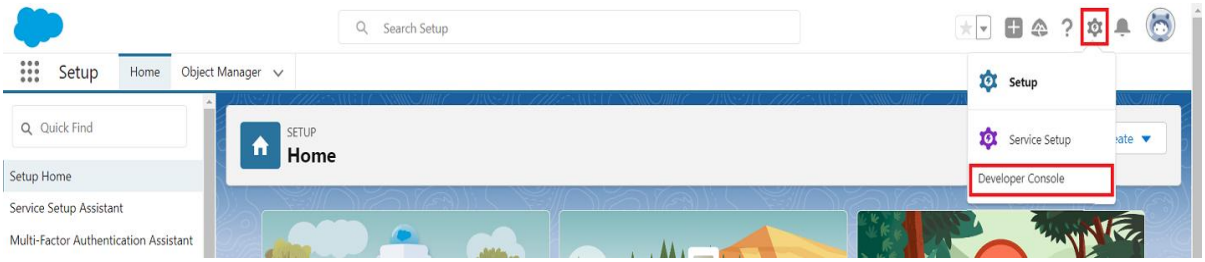
```
public class RecordDeletions Implements Database.Batchable<Sobject>{
    public Database.QueryLocator start(Database.BatchableContext bc) {
         string query = 'SELECT Id, Name, WorkOrder_ID__c, Technician_ID__c, Assignment_Date__c, Completion_Date__c FROM Assignment__c WHERE Completion_Date__c = 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){
    }
}
```

```
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.Batchab leContext bc){
    }
}
```

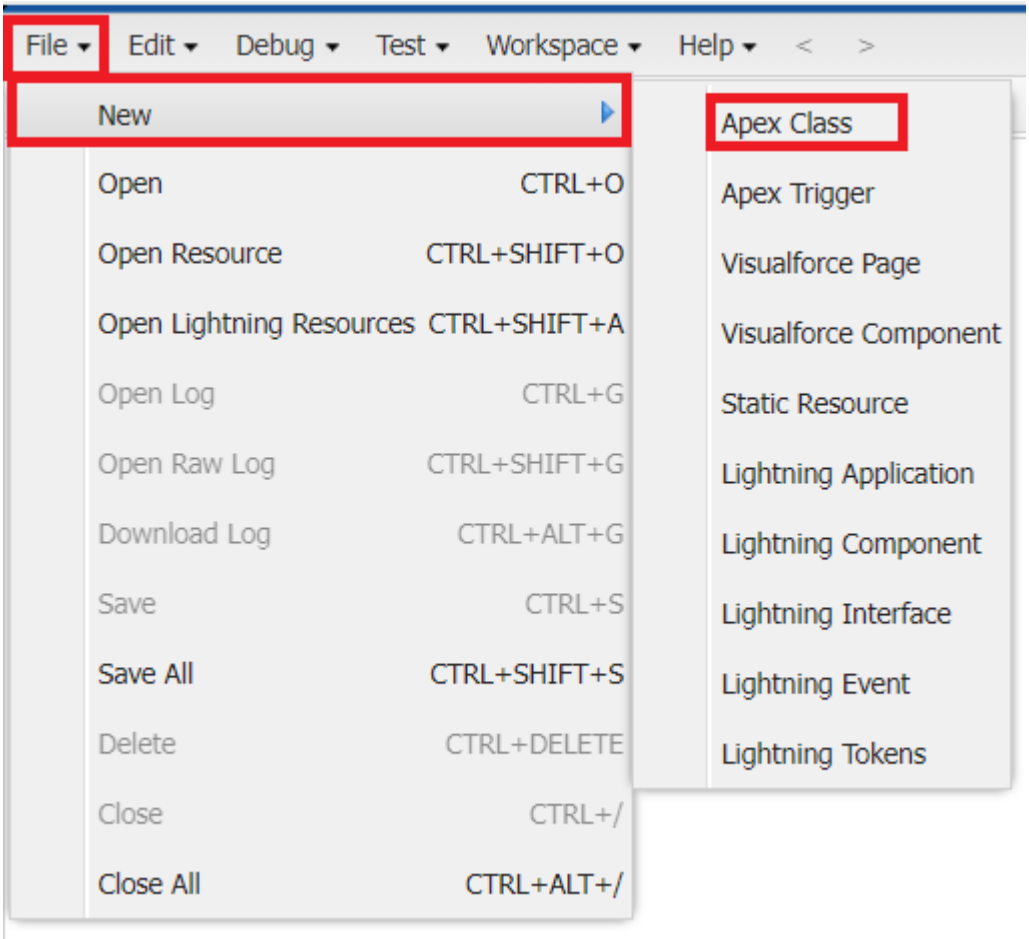
7. Save the code.(click on file --> Save)

ACTIVITY 8 : Create an Apex Schedule Class

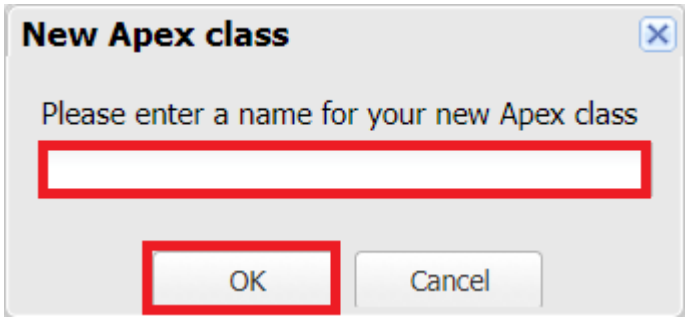
1.Go to Setup --> Click on the gear icon --> Select Developer Console.



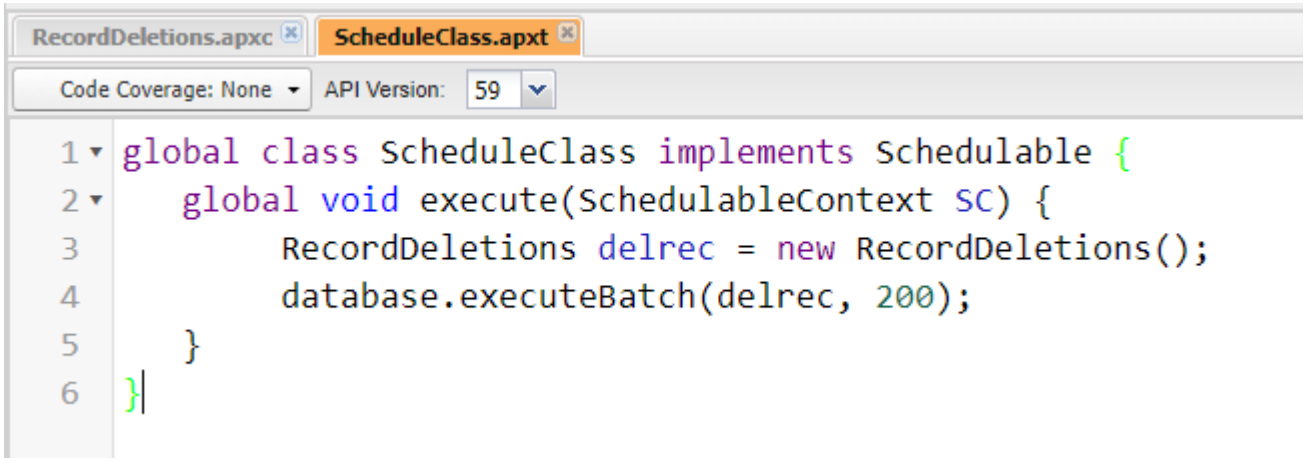
2.Then we can see the Developer console. Click on the developer console and you will navigate to a new console window.
3.To create a new Apex Class follow the below steps:
Click on the file --> New --> Apex Class.



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);
  }
}
```

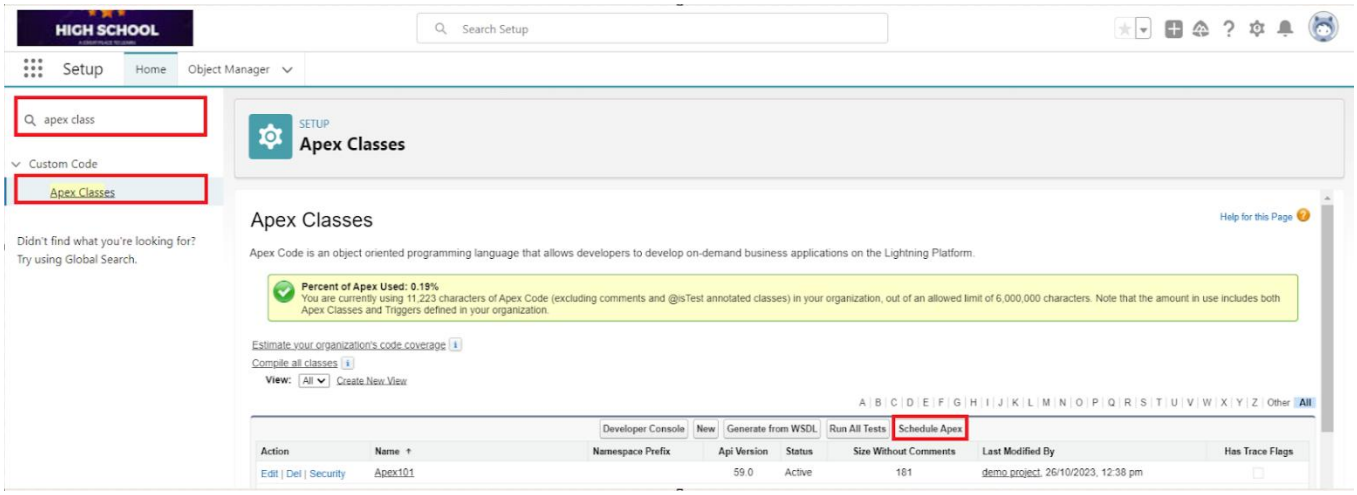
7.Save the code.(click on file ? Save)

Field service workorder optimization

ACTIVITY 9 : Create a Schedule Apex

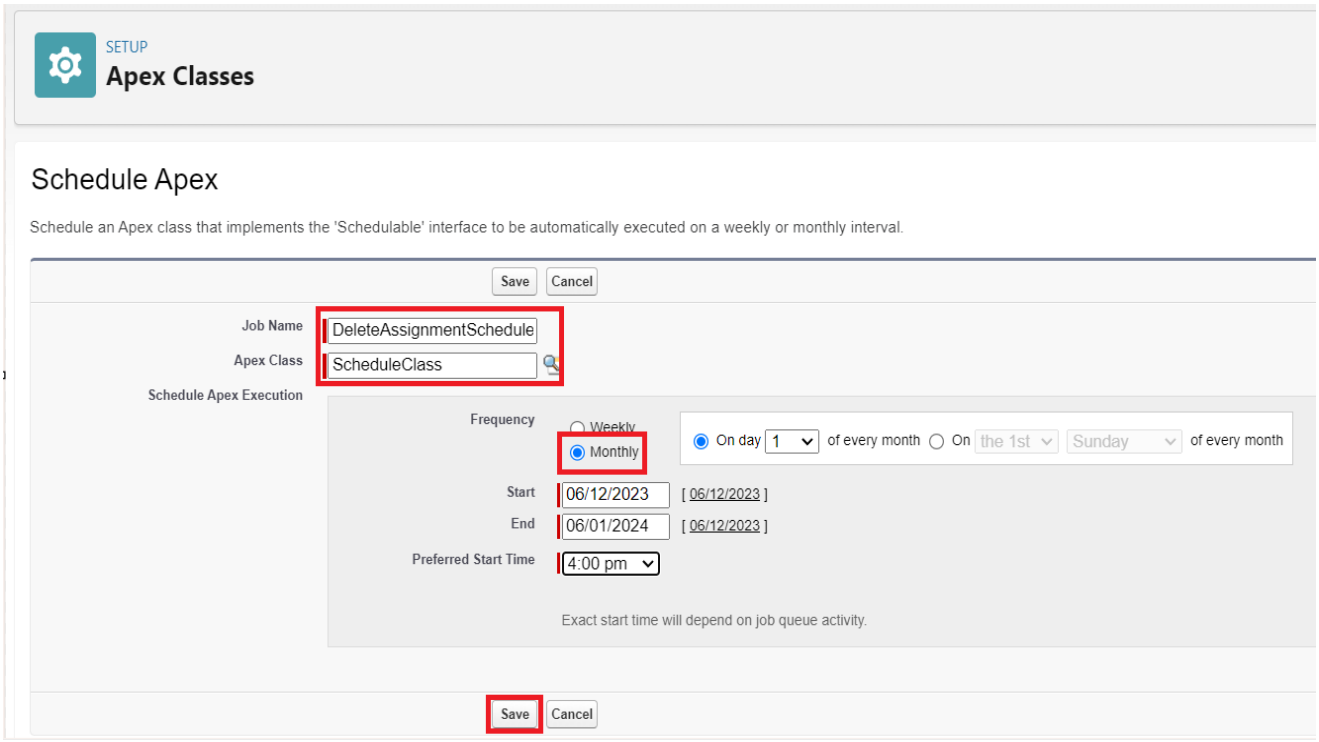
Schedule the Apex class:

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.

a. Job Name : DeleteAssignmentSchedule
b. Apex Class : ScheduleClass (from clicking on lookup icon)
c. Frequency : Monthly
d. Preferred Start Time : Select any time



4.Click Save.

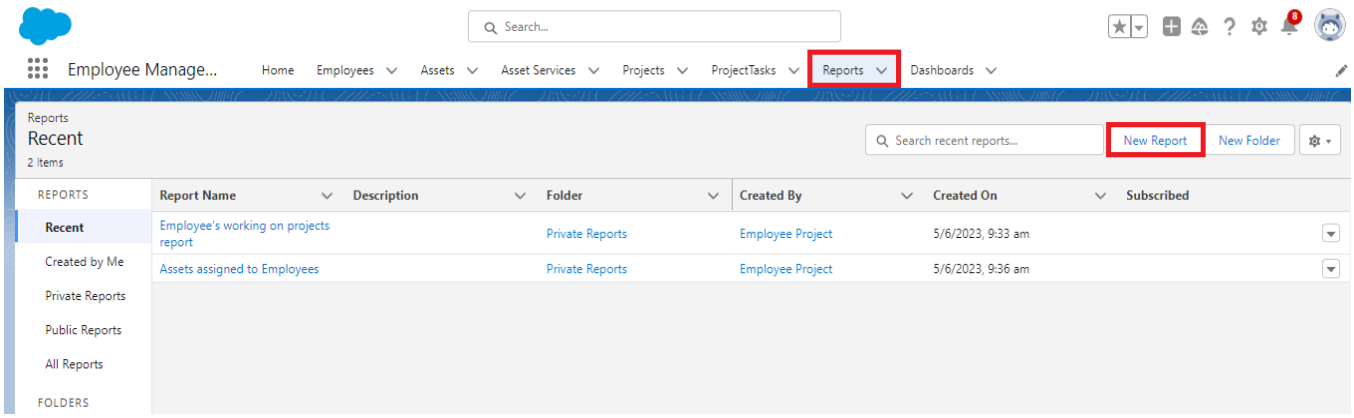
Field service workorder optimization

Reports & Dashboards

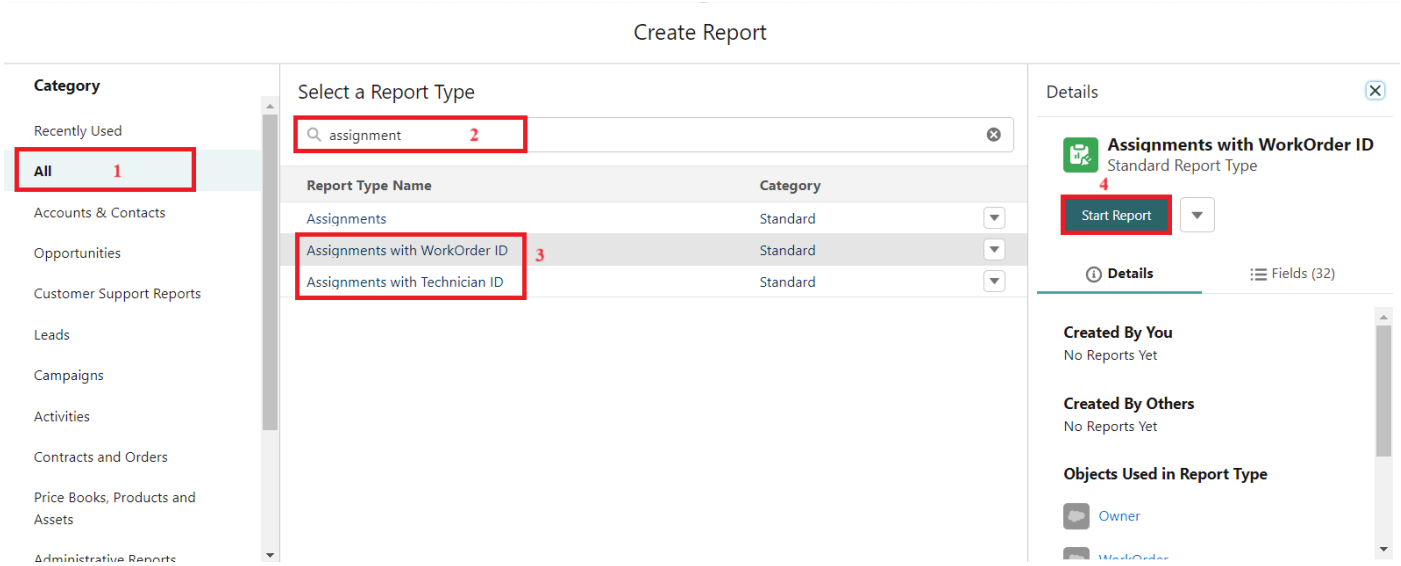
Salesforce Reports and Dashboards are powerful tools that empower users to visualize and analyze data within the Salesforce platform. They play a crucial role in providing insights, monitoring performance, and making informed business decisions.

ACTIVITY 1 : Report

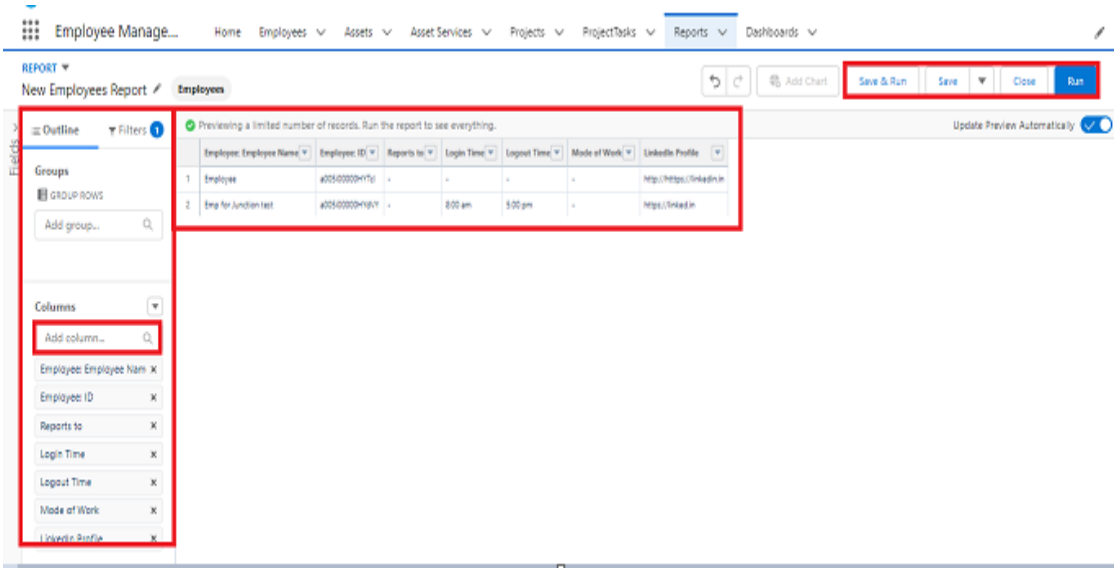
1. Go to the app --> click on the reports tab
2. Click New Report.



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.

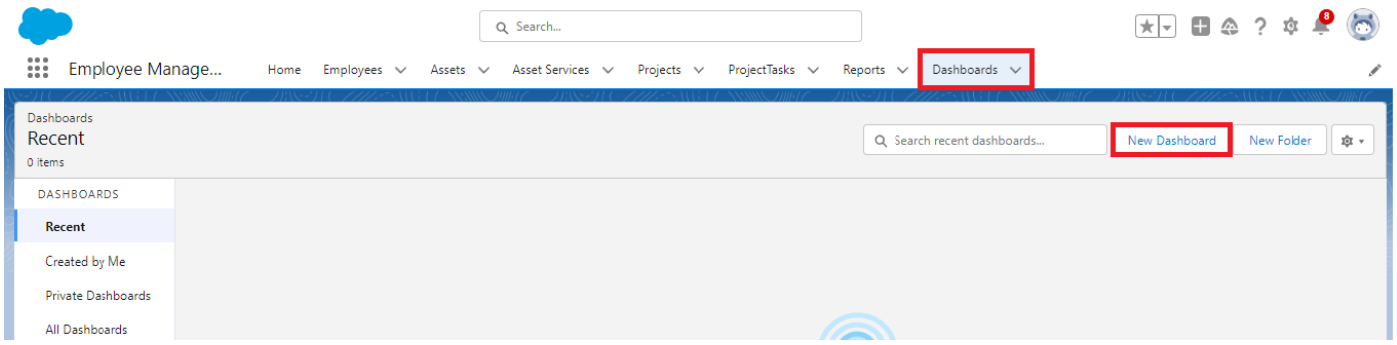
ACTIVITY 2 : Create Reports

1. Create a report with report type: “WorkOrders Status Reports”.
2. Create a report with report type: “Technician and Assignment Details Reports”.

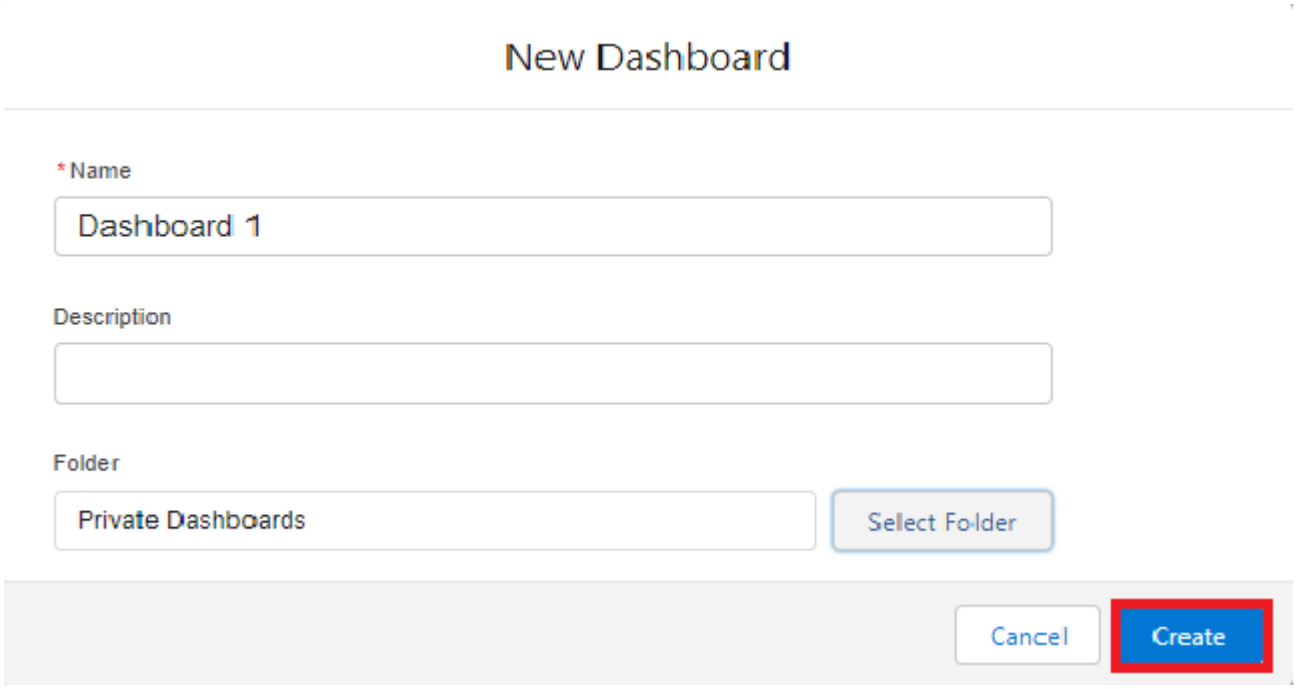
Field service workorder optimization

ACTIVITY 3 : Dashboard

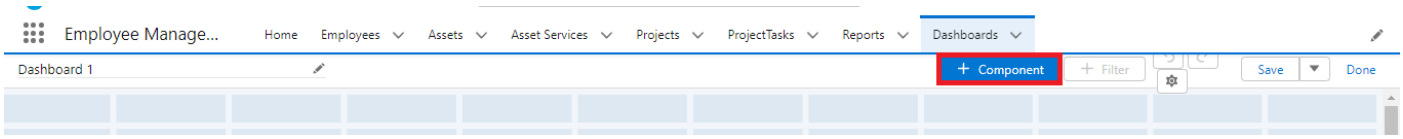
1. Go to the app --> click on the Dashboards tabs.



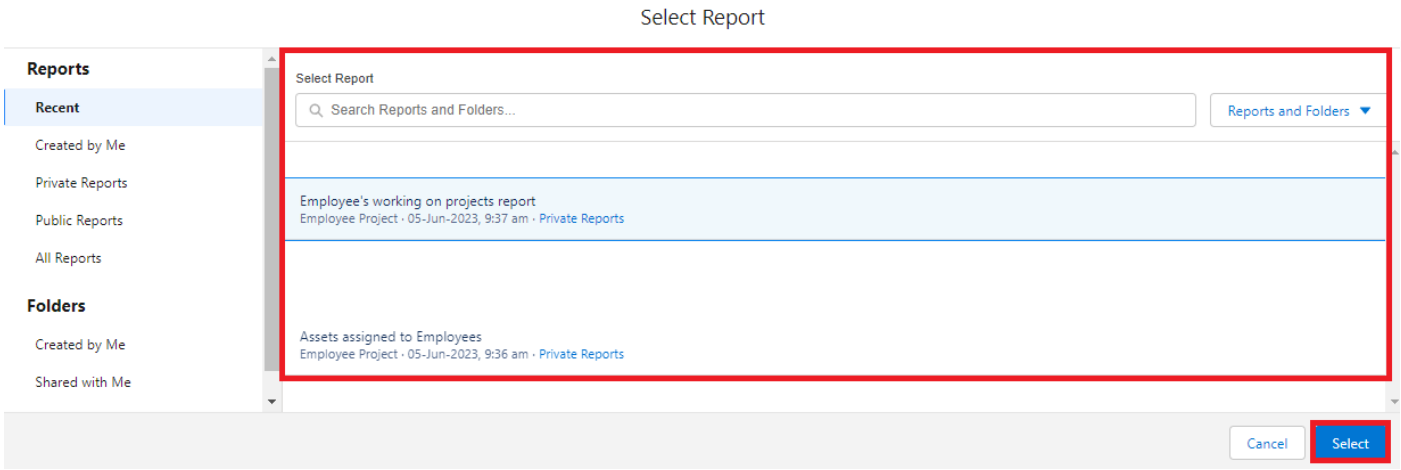
2. Give a Name and click on Create.



3.Select add component.



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



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

ACTIVITY 4 : Create Dashboards

Create another Dashboard as we discussed in activity 3 which shows the details of completed workorder status in a vertical bar graph.

Field service workorder optimization