



APEX TRIGGER EXAMPLES





Example 1

Write a trigger on Account, when an account is inserted, automatically account billing address should populate into the account shipping address.

trigger **changeshippingaddress** on **Account** (**before insert**) {

```
for(Account a : Trigger.new){
```

```
//We should first check whether the user is passing  
//null data
```

```
if(a.BillingStreet!=null){  
    a.ShippingStreet=a.BillingStreet;  
}
```

```
if(a.BillingCity!=null){  
    a.ShippingCity=a.BillingCity;  
}
```

```
if(a.BillingState!=null){  
    a.ShippingState=a.BillingState;  
}
```

```
if(a.BillingPostalCode!=null){  
    a.ShippingPostalCode=a.BillingPostalCode;  
}
```

```
if(a.BillingCountry!=null){  
    a.ShippingCountry=a.BillingCountry;  
}
```

```
}
```

```
}
```

Test Case:

1. Go to Accounts and create a new account. Populate the billing address fields as shown in the below image and leave the shipping address fields blank as shown in the below image. Click save.

The screenshot shows the 'Address Information' section of the Salesforce Account creation form. The 'Billing Address' group is highlighted in yellow, containing the following data:

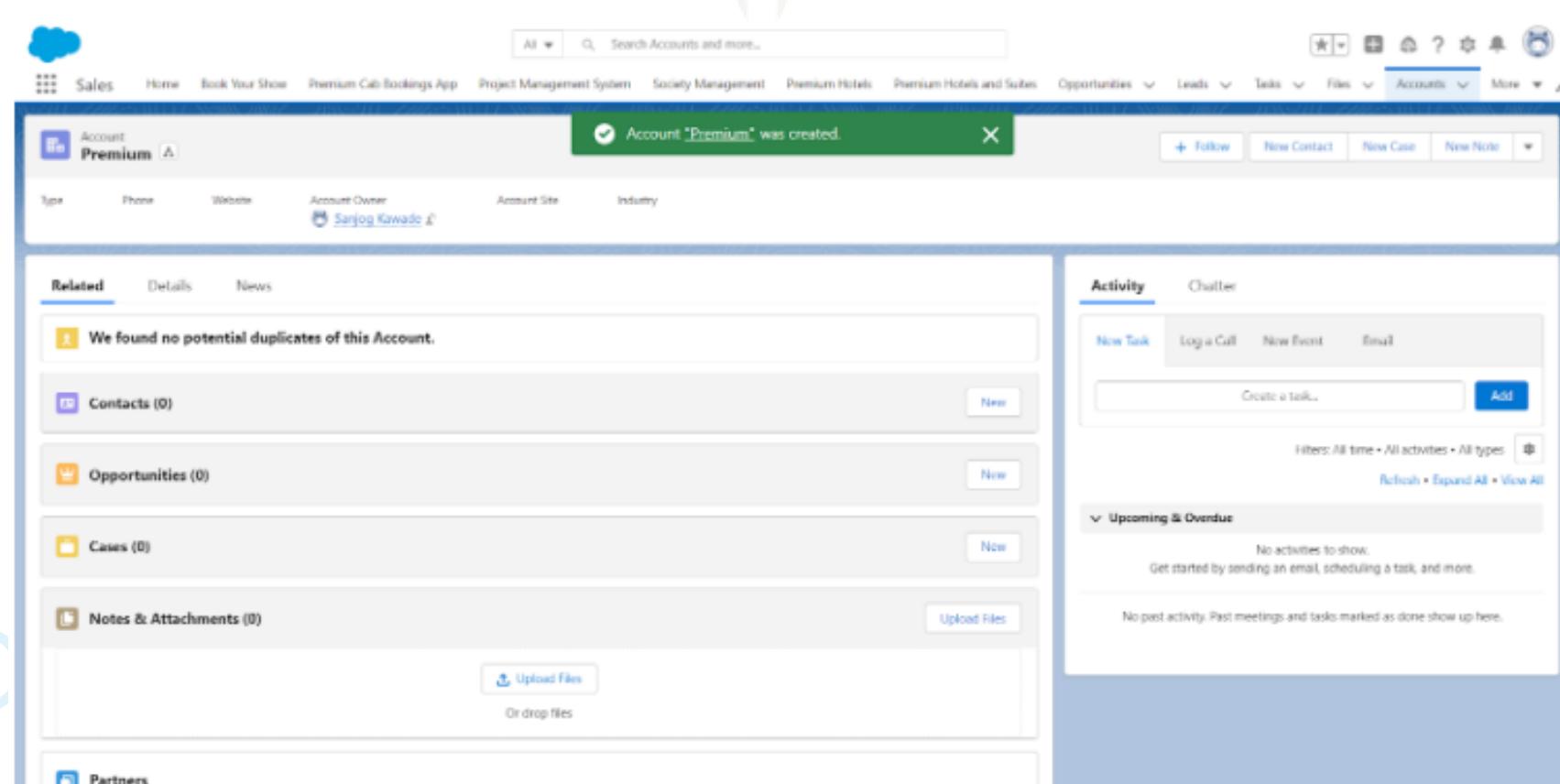
- Billing Street: JM Road
- Billing City: Pune
- Billing State/Province: Maharashtra
- Billing Zip/Postal Code: 411004
- Billing Country: India

The 'Shipping Address' group is shown below, with all fields empty:

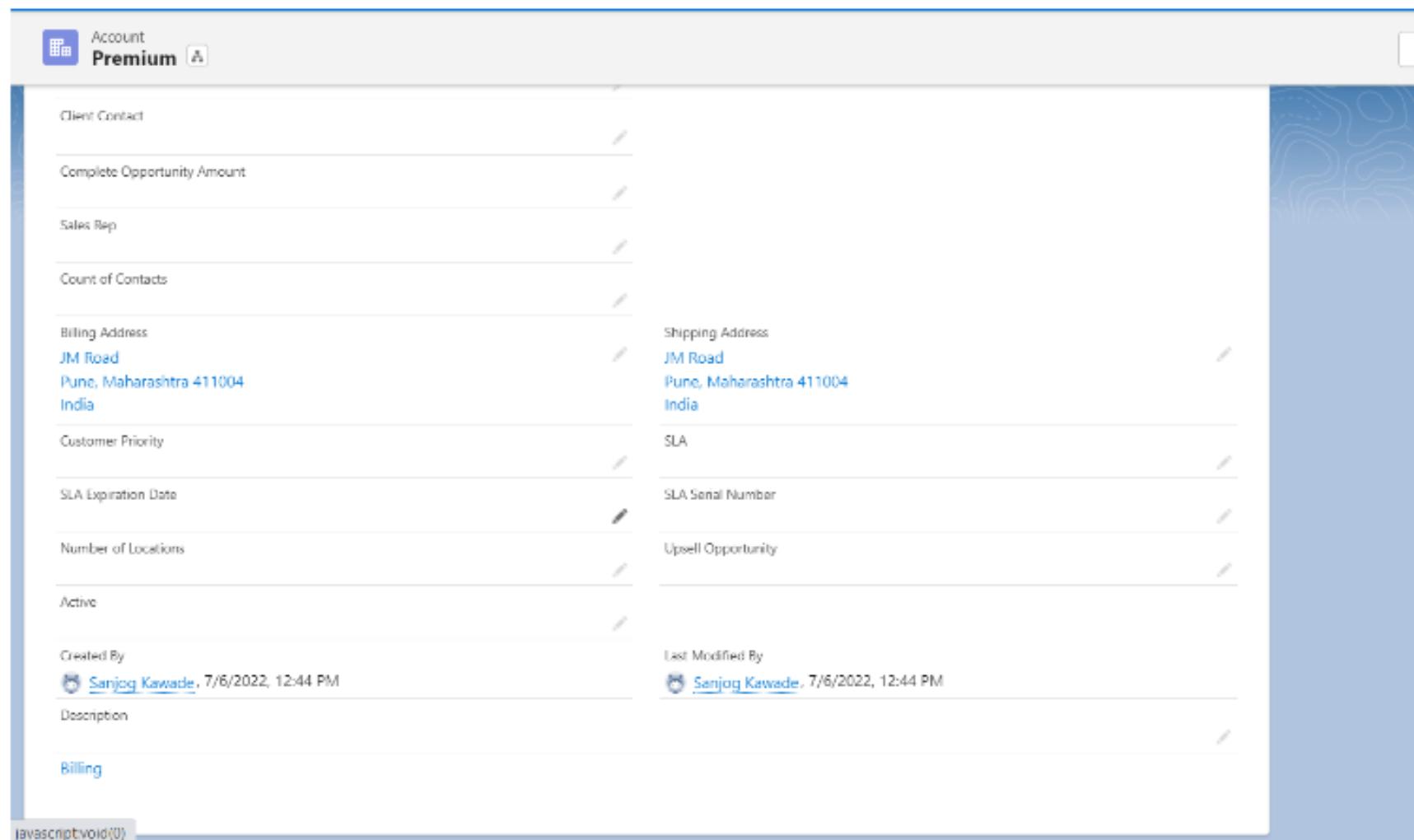
- Shipping Street:
- Shipping City:
- Shipping State/Province:
- Shipping Zip/Postal Code:
- Shipping Country:

Below the address sections, there is an 'Additional Information' section with dropdowns for Customer Priority (set to '--None--') and SLA (set to '--None--'). At the bottom right of the form are three buttons: 'Cancel', 'Save & New', and a blue 'Save' button.

2. The record is now saved as shown below image. Click on the details page and check the shipping address which we kept blank while creating the record.



- 3. The account billing address we entered is populated in the shipping address as well.





Example 2

Write a trigger on the Account when the Account is updated check all opportunities related to the account. Update all Opportunities Stage to close lost if an opportunity created date is greater than 30 days from today and stage not equal to close won.

```
trigger OppoStageUpdate on Account (after update){

    Set<Id> accountIds = new Set<Id>();

    for(Account a:Trigger.new){
        accountIds.add(a.Id);
    }

    //day30 is the date which is 30 days less than
    //today
    DateTime day30=system.now()-30;
    List<Opportunity> oppListToUpdate=new List<Opportunity>();

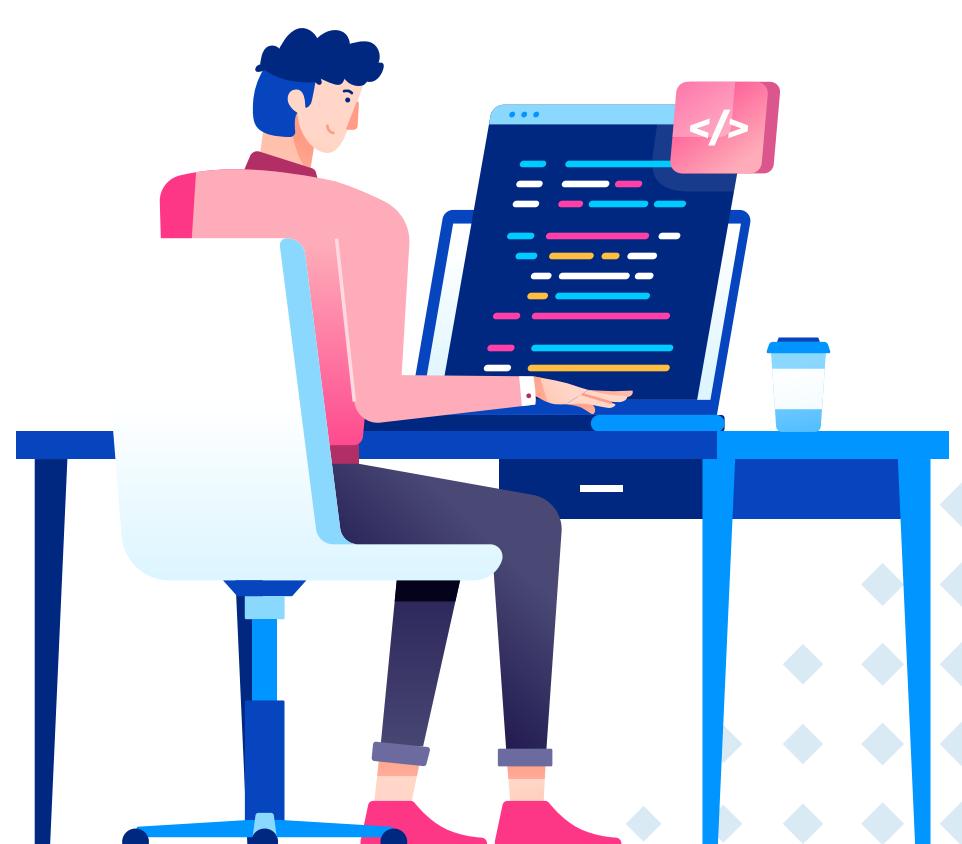
    //getting the opportunities whose account has
    //been updated
    List<Opportunity> oppList = [Select Id, AccountId,
        StageName,CreatedDate, CloseDate from Opportunity where
        AccountId in :accountIds];

    if(oppList.size()>0)
    {
        for(Opportunity opp : oppList)
        {

            //checking for condition if created date is
            //greater than 30 days from today and stage not equal to
            //close won

            if(opp.CreatedDate<day30 && opp.StageName!=
                'Closed Won')
            {
                opp.StageName='Closed Lost';
            }
        }
    }
}
```

```
//This is a mandatory field when we update the  
//CloseDate  
  
    opp.CloseDate=system.today();  
    oppListToUpdate.add(opp);  
//putting the changed opportunity to separate list  
//to update later  
}  
}  
}  
  
//checking if the opportunity list which has  
//changed is having records or not  
  
if(oppListToUpdate.size()>0){  
    update oppListToUpdate;  
}  
}
```





Test Case:

1. Select an account with opportunities created date greater than 30 days.
2. Make sure at least one related opportunity of the account is not closed won or closed lost.

3. Now update the account details with your choice of update (here we change the rating from hot to cold)



Account Details

Edge Communications

Related **Details** **News**

Account Owner: Sanjog Kawade

Account Name: Edge Communications

Parent Account

Account Number: CD451796

Account Site

Type: Customer - Direct

Industry: Electronics

Annual Revenue: \$139,000,000

Premium

out of business

No of open opportunities

Recent Types

Activity **Chatter**

New Task **Log a Call** **New Event** **Email**

Create a task... **Add**

Filters: All time • All activities • All types **Refresh** **Expand All** **View All**

Upcoming & Ongoing

No activities to show.

Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

4. Now check the related list again, and all the related opportunities (which are not closed won or closed lost) will become closed lost. As shown below image.

Account **Edge Communications**

Related **Details** **News**

We found no potential duplicates of this Account.

Contacts (2)

- Sean Forbes: Title: CFO, Email: sean@edge.com, Phone: (512) 757-6000
- Rose Gonsalez: Title: SVP, Procurement, Email: rose@edge.com, Phone: (512) 757-6000

Opportunities (4)

Opportunity Name	Stage	Amount	Close Date
Edge Emergency Generator	Closed Lost	\$75,000.00	7/6/2022
Edge SLA	Closed Lost	\$60,000.00	7/6/2022
Edge Installation	Closed Won	\$50,000.00	2/1/2022

Activity **Chatter**

New Task **Log a Call** **New Event** **Email**

Create a task... **Add**

Filters: All time • All activities • All types **Refresh** **Expand All** **View All**

Upcoming & Ongoing

No activities to show.

Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.



Example 3

Once an Account is inserted an email should go to the System Admin user with specified text below.
An account has been created and the name is “Account Name”.

```

trigger sendEmailToAdmin on Account (after insert){
    //To send list of mails when there are insertion
    //of list of Accounts

List<Messaging.SingleEmailMessage> mails = new
List<Messaging.SingleEmailMessage>();

//Query to get the Email of a System
//Administrator

User userObj=[select Id,Profile.Name,Email from user where
            Profile.Name='System Administrator'];
for(Account accObj:Trigger.new)
{
    //Checking if the user email is not null
    if(userObj.Email!=null)
    {
        //Assigning a single Mail to send
        Messaging.SingleEmailMessage mail = new
        Messaging.SingleEmailMessage();

        //Assigning the Sender Name for Mail
        mail.setSenderDisplayName('Salesforce');

        //We are make all this below fields as false
        //because those are not needed for now
        mail.setUseSignature(false);
        mail.setBccSender(false);
        mail.setSaveAsActivity(false);
}

```

```
//We are make all this below fields as false  
//because those are not needed for now  
    mail.setUseSignature(false);  
    mail.setBccSender(false);  
    mail.setSaveAsActivity(false);  
  
//Assigning the receiver Mail Address  
    mail.toAddresses = new String[]{userObj.Email};  
  
//Assign the Subject of the Mail  
    mail.setSubject('New Account was Created.');//A variable to write the body of the Mail  
    String body = 'Dear System Administrator,<br/>';  
    body += 'An account has been created and name is  
    '+accObj.Name+'.';  
  
//Assigning the variable in which we wrote the  
//body to the Mail Body  
    mail.setHtmlBody(body);  
  
//Adding each single mail to be sent to the list  
//of mails  
    mails.add(mail);  
}  
}  
  
//Checking if the list of mails is not empty  
if(mails.size()>0){  
// "Messaging.sendEmail(mails)" is used to send  
//the list of mails
```

```
Messaging.SendEmailResult[] results =  
Messaging.sendEmail(mails);  
//we are checking if the mails are sent or not.  
if (results[0].success)  
{  
    System.debug('The email was sent successfully.');//  
}  
else  
{  
    System.debug('The email failed to send:  
    '+ results[0].errors[0].message);  
}  
}  
}
```





Test Case:

1. Create a new account, give it a suitable name and save.

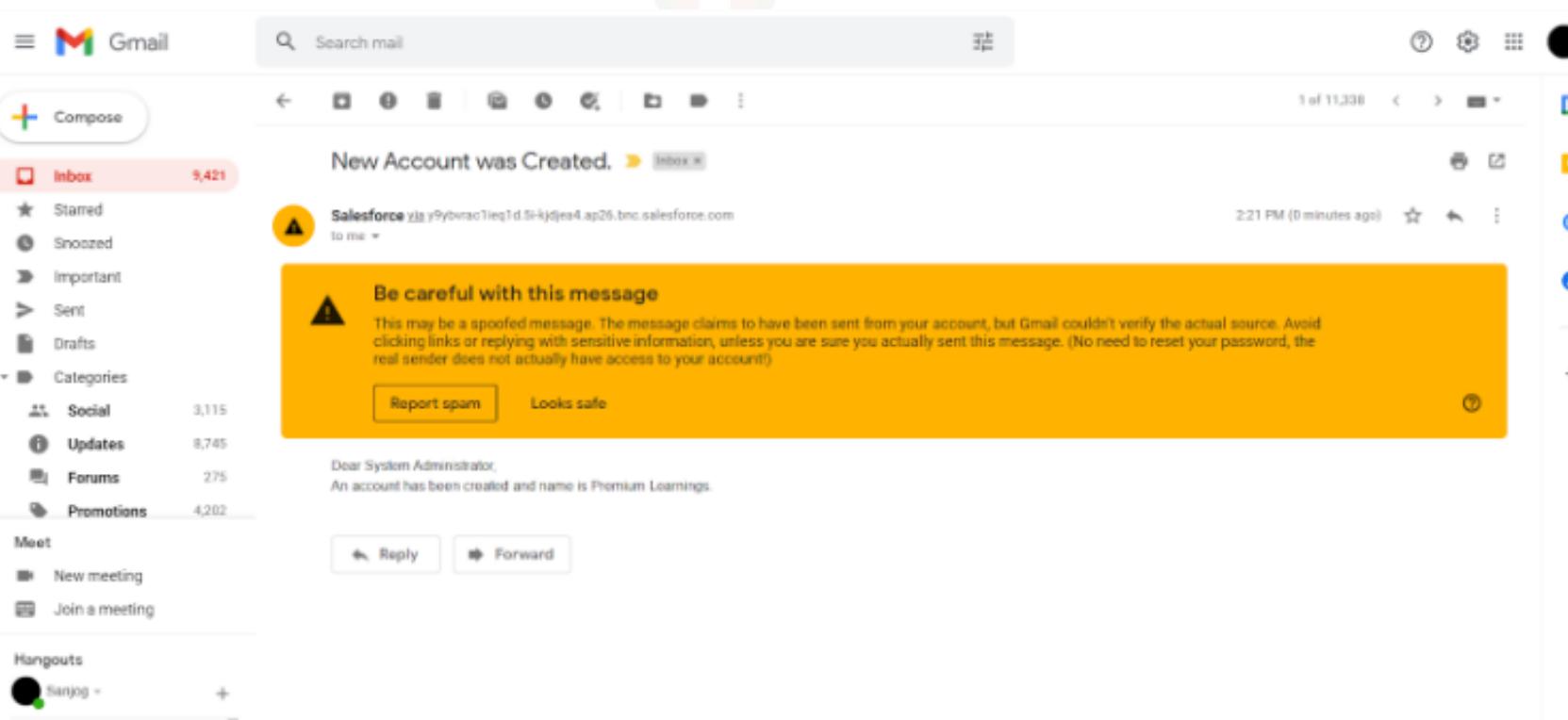
New Account

Account Information

Account Owner Sanjog Kawade	Rating Hot
* Account Name Premium Learnings	Phone
Parent Account Search Accounts...	Fax
Account Number	Website
Account Site	Ticker Symbol
Type --None--	Ownership --None--
Industry --None--	Employees
Annual Revenue	SIC Code
Premium <input type="checkbox"/> out of business	

Buttons: Cancel, Save & New, Save

2. Now check your email (you being the system administrator of your salesforce org). you will receive a mail with the format as shown below.



Example 4

Once an Account will update then that Account will update with the total amount from All its Opportunities on the Account Level. The account field name would be " Total Opportunity Amount ".create a field named Total Opportunity Amount of datatype number

```
trigger totalAmountofRelatedOpp on Account (before update)
{
    //Get the Account Id's of the List of Accounts
    //getting updated to get its related Opportunities

    Set<Id> accId=new Set<Id>();
    for(Account acc:Trigger.new)
    {
        //before making changes see to that "Total
        //Opportunity Amount" field is 0
        acc.Total_Opportunity_Amount__c=0;
        accId.add(acc.Id);
    }

    //map to get the Account Id and the sum of its
    //related opportunities amount to insert later
    Map<Id,Double> amountMap = new Map<Id,Double>();

    //AggregateResult to get the sum of opportunities
    //amount grouped by Account Id
    List<AggregateResult> results=[select
        AccountId,sum(Amount)TotalAmount from opportunity where
        AccountId in :accId group by AccountId];
    if(results.size()>0){
        for(AggregateResult a: results){

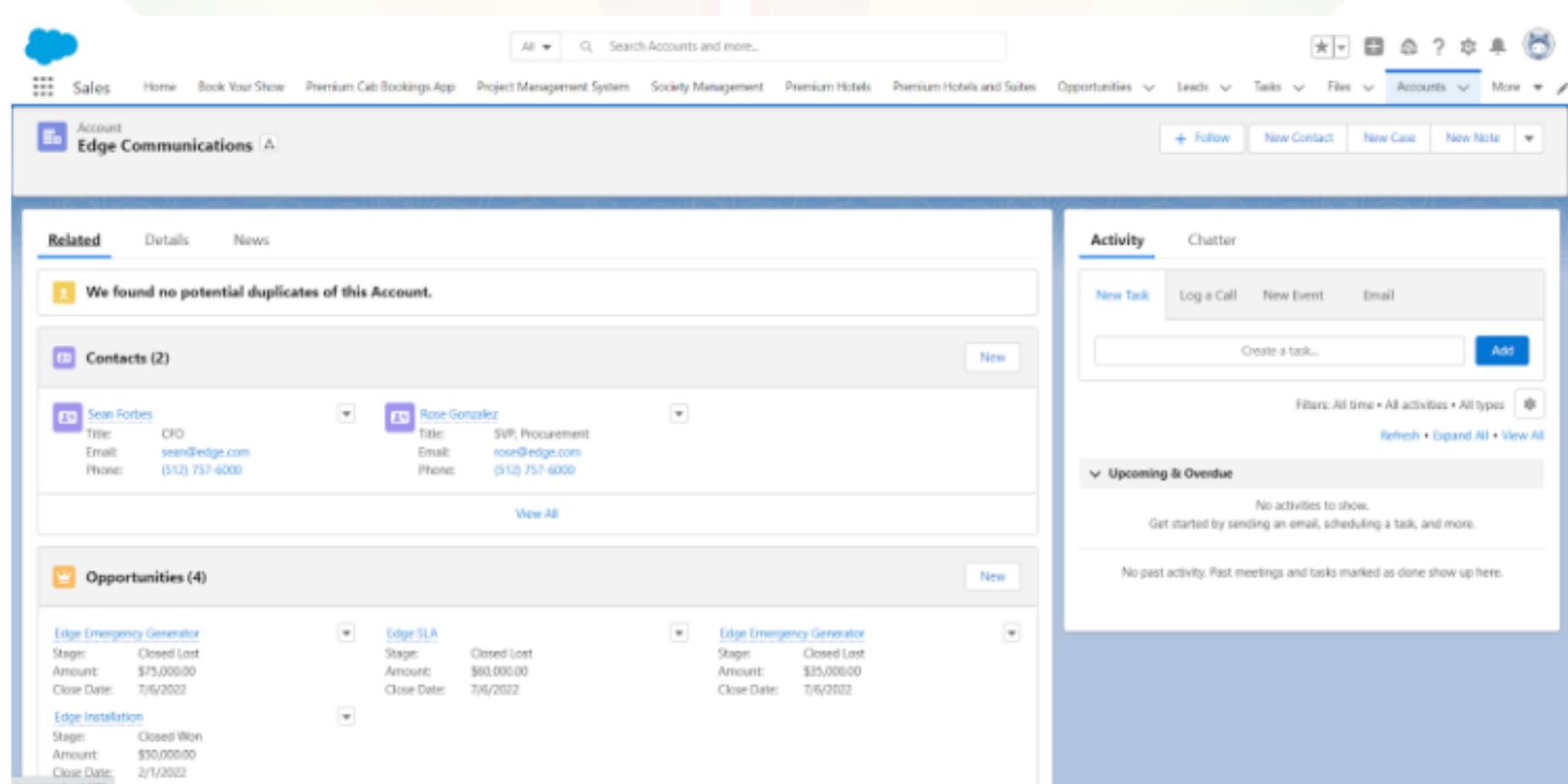
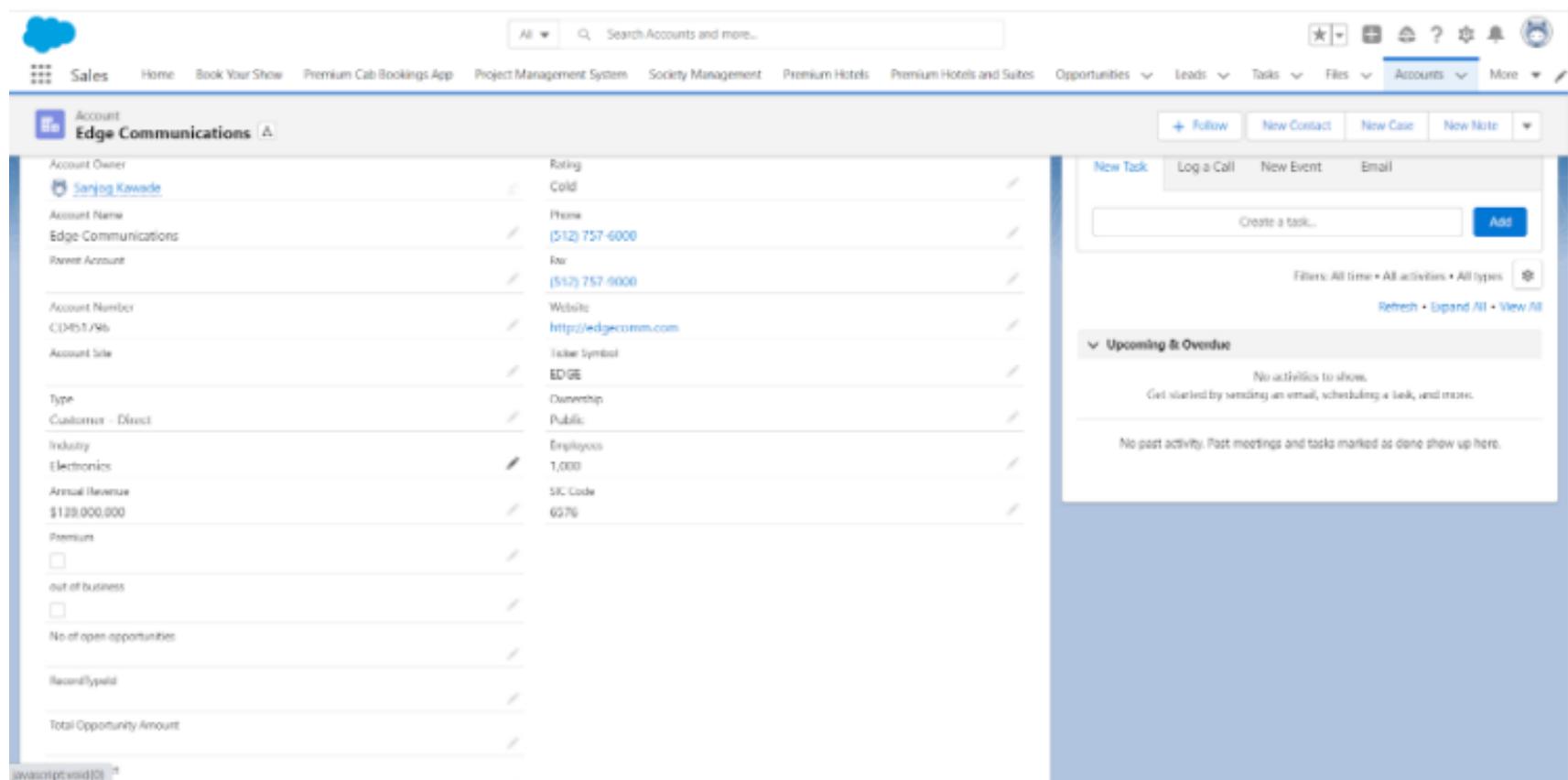
```

```
//getting the AccountId and sum(Amount) in  
//separate variables and putting it to map  
    Id accountId = (Id)a.get('AccountId');  
    double TotalAmount = (double)a.get('TotalAmount');  
    amountMap.put(accountId,TotalAmount);  
}  
}  
  
//Again looping the accounts which are getting  
//updated and making changes in "Total  
//Opportunity Amount" field  
for(Account acc:Trigger.new)  
{  
    if(amountMap.containsKey(acc.Id))  
    {  
        acc.Total_Opportunity_Amount__c=amountMap.get(acc.Id);  
    }  
}  
}
```

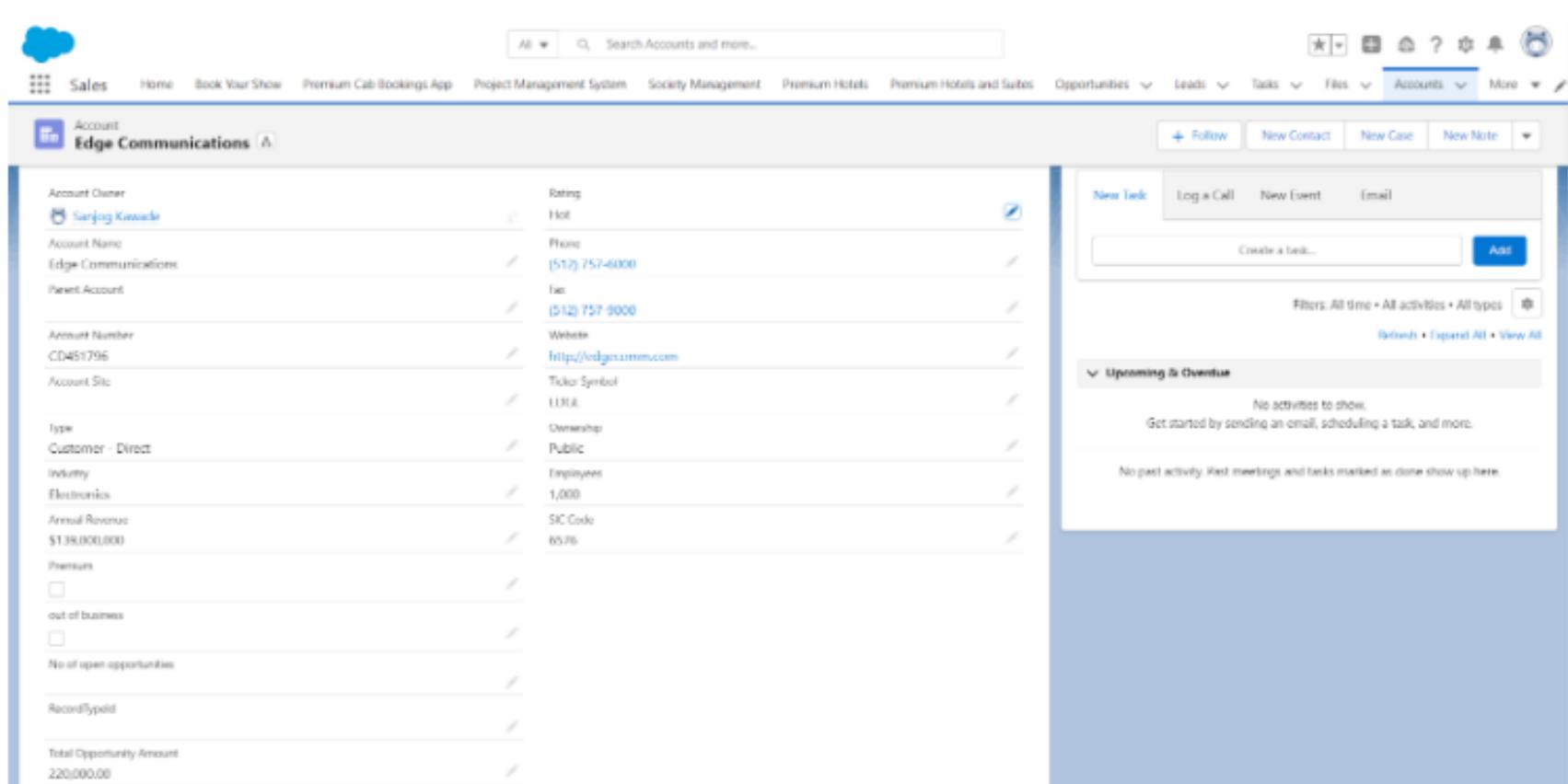


Test Case:

1. Select an account with existing related opportunities (or create one). Notice that the total opportunity amount field is blank.



2. Now update the account (change rating field). The apex trigger will be fired and total opportunity amount field will be populated with amount of all opportunities added as shown in figure.





Example 5

Create a field on Account Named (Client Contact lookup to Contact). Once an Account is inserted a Contact will create with the name of the Account and that Contact will be the Client Contact on the Account.

```

trigger updateClientContactFieldOnAccount on Account
(after insert){

    //Account list to update after the doing changes
    List<Account> accListUpdate = new List<Account>();

    //Contacts to insert which has same name as
    //Account Name
    List<Contact> conList = new List<Contact>();

    //Account Id list to update later after changes
    //are done
    Set<Id> accId = new Set<Id>();

    //Map of Account Id and the Account record
    Map<Id,Account> mapVariable = new Map<Id,Account>();

    for(Account accObj : Trigger.new)
    {

        //Contact which has to be inserted with same name
        //as Account and putting that Contact as related
        //Contact
        Contact conObj =new Contact();
        conObj.LastName = accObj.Name;
        conObj.AccountId = accObj.Id;
        conList.add(conObj);
        accId.add(conObj.AccountId);
    }
    if(conList.size()>0){
        insert conList;
    }
}

```

```
//Getting list of Accounts to put into Map
List<Account> accList =[Select Id,Client_Contact__c from
                           Account where Id=:accId];

if(accList.size()>0){
    for(Account acc:accList){
        mapVariable.put(acc.Id,acc);
    }
}

if(conList.size()>0){
/*Looping inserted Contact and Check whether the Account
   Id is related and get its Account record from map and
   change the value of the Client Contact with the current
   looping Contact*/
    for(Contact cObj:conList){
        if(mapVariable.containsKey(cObj.AccountId)){
            Account aObj=mapVariable.get(cObj.AccountId);
            aObj.Client_Contact__c=cObj.Id;
            accListUpdate.add(aObj);
        }
    }
}

//Update the Account in which chnages has been done
if(accListUpdate.size()>0){
    update accListUpdate;
}
```



Test Case:

1. Create an account record as shown in figure. Save it

New Account

Account Information

Account Owner Sanjog Kawade	Rating Hot
* Account Name Premium Learnings	Phone
Parent Account Search Accounts...	Fax
Account Number	Website
Account Site	Ticker Symbol
Type --None--	Ownership --None--
Industry --None--	Employees
Annual Revenue	SIC Code
Premium <input type="checkbox"/> out of business	

Cancel Save & New Save

2. The record is saved.

Cloud icon

All Search Accounts and more... Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Accounts More

Account Premium Learnings

Type	Phone	Website	Account Owner	Account Site	Industry
Sanjog Kawade					

Related Details News

Account Owner Sanjog Kawade	Rating Hot
Account Name Premium Learnings	Phone
Parent Account	Fax
Account Number	Website
Account Site	Ticker Symbol
Type	Ownership
Industry	Employees
Annual Revenue	SIC Code
Premium <input type="checkbox"/> out of business	

Activity Chatter

New Task Log a Call New Event Email
Create a task... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

3. Check the related lists of the record. You will see a contact record created with the same name as the account.

The screenshot shows the Salesforce Account page for 'Premium Learnings'. At the top, there's a navigation bar with links like Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, and More. Below the navigation is a search bar with placeholder text 'Search Accounts and more...'. The main content area displays the account details for 'Premium Learnings', including fields for Type, Phone, Website, Account Owner (Sanjog Kawade), Account Site, and Industry. A 'Related' section contains tabs for Related, Details, and News. Under the 'Related' tab, it says 'We found no potential duplicates of this Account.' and lists 'Contacts (1)', 'Opportunities (0)', 'Cases (0)', and 'Notes & Attachments (0)'. To the right, there's an 'Activity' section with tabs for Activity and Chatter. The Activity tab shows buttons for New Task, Log a Call, New Event, and Email, along with a 'Create a task...' input field and an 'Add' button. Below these are sections for 'Upcoming & Overdue' and 'Past activity'. The 'Upcoming & Overdue' section indicates 'No activities to show.' and 'Get started by sending an email, scheduling a task, and more.' The 'Past activity' section says 'No past activity. Past meetings and tasks marked as done show up here.'

Example 6

Write a trigger on contact to prevent duplicate records based on Contact Email & Contact Phone

```
trigger dupEmailPhone on Contact (before insert,  
before update)  
{  
    Map<String, Contact> emailMap = new Map<String, Contact>();  
    Map<String, Contact> phoneMap = new Map<String, Contact>();  
  
    for(Contact contact : trigger.new){  
        if(trigger.isInsert){  
            emailMap.put(contact.Email, contact);  
            phoneMap.put(contact.Phone, contact);  
        }  
        if(trigger.isUpdate){  
            if(trigger.oldMap.get(contact.Id).Email!=contact.Email)  
            {  
                emailMap.put(contact.Email, contact);  
            }  
            if(trigger.oldMap.get(contact.Id).Phone!=contact.Phone)  
            {  
                phoneMap.put(contact.Phone, contact);  
            }  
        }  
    }  
  
    String errorMessage = '';
```

```

//getting the Contacts whose Email or Phone
//already exists

List<Contact> existingContactList = [Select Id, Email,
    Phone FROM Contact Where Email IN:emailMap.keySet() OR
    Phone IN:phoneMap.keySet()];

if(existingContactList.size() > 0){
    for(Contact contactRec : existingContactList){
        if(contactRec.Email != null){
            if(emailMap.get(contactRec.Email) != null){
                errorMessage='Email ';
            }
        }
        if(contactRec.Phone != null){
            if(phoneMap.get(contactRec.Phone) != null) {
                errorMessage = errorMessage +(errorMessage != '' ?
                    'and Phone ' : 'Phone ');
            }
        }
        if(errorMessage != ''){
            trigger.new[0].addError('Your Contact'+errorMessage
                +'already exists in system.');
        }
    }
}

```



Test Case:

1. Create a contact record, with the email field populated(remember this email). Click save.

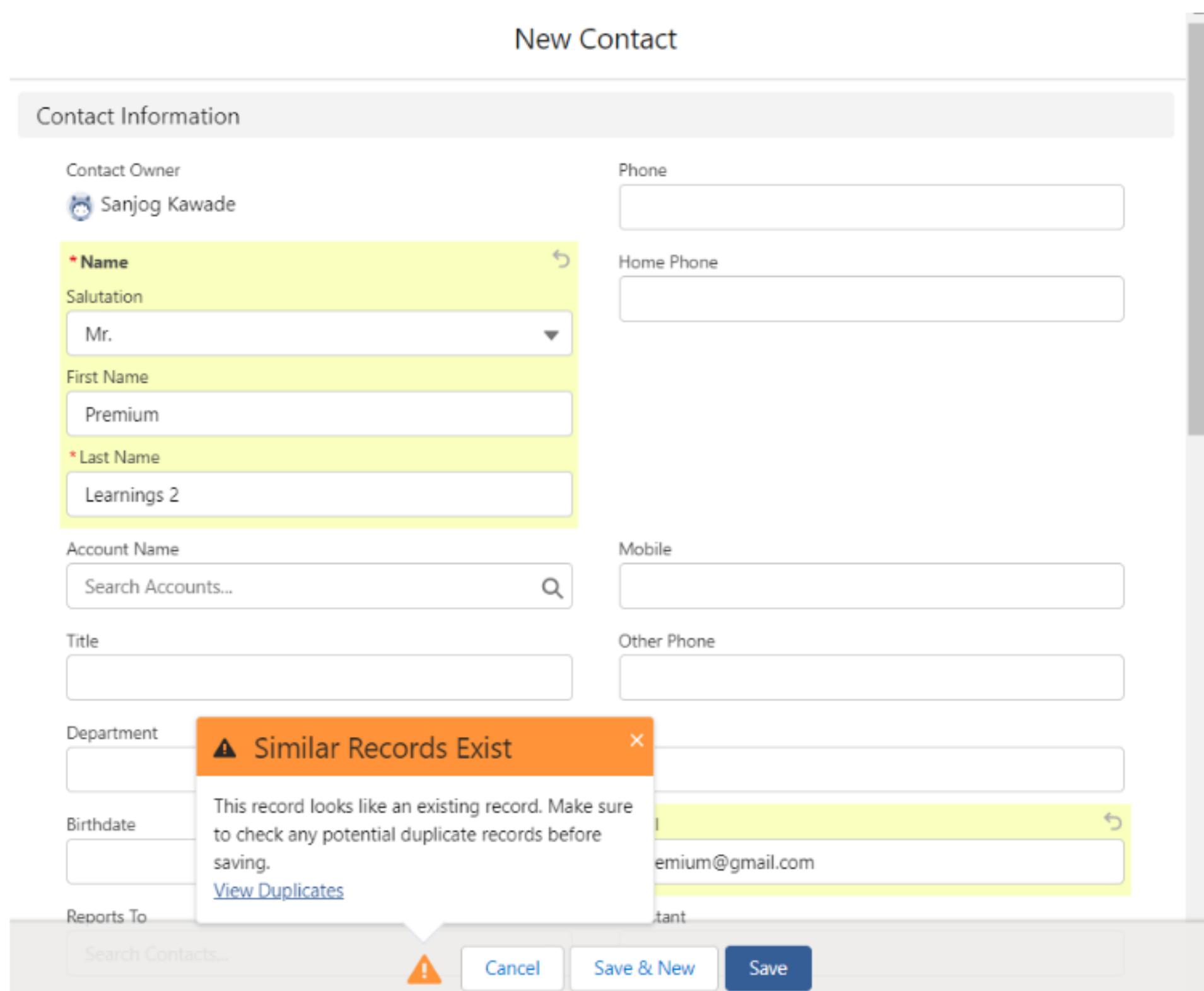
New Contact

Contact Information

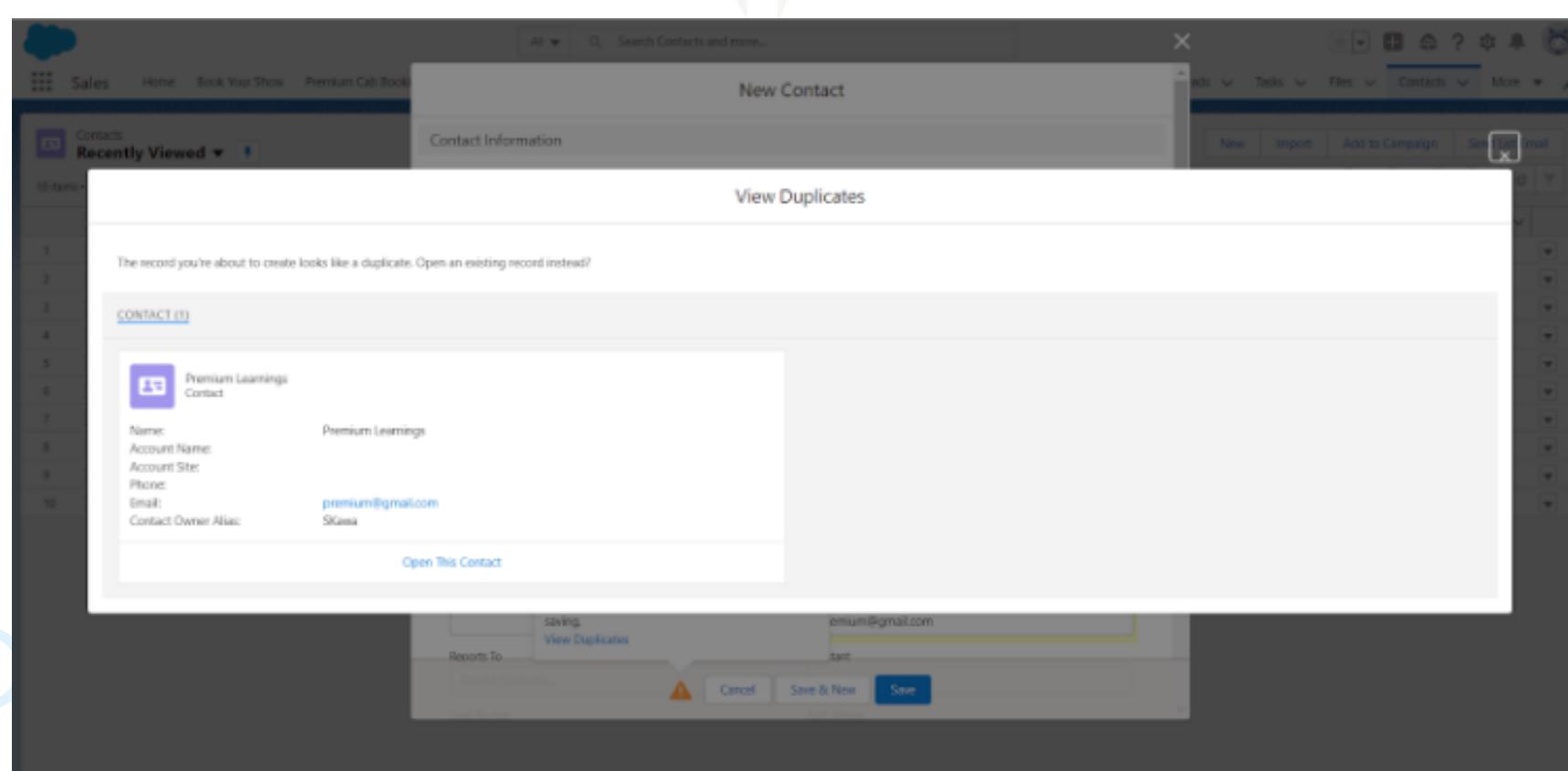
Contact Owner  Sanjog Kawade	Phone
* Name Salutation Mr.	Home Phone
First Name Premium	
* Last Name Learnings	
Account Name Search Accounts...	Mobile
Title	Other Phone
Department	Fax
Birthdate	Email premium@gmail.com
Reports To Search Contacts...	Assistant

Cancel Save & New Save

2. Now create another contact with same email. When you click save, the following error message is displayed and record fails to save.



3. When you click on View Duplicates, the previously created contact record with the same email is shown.



Example 7

Write a trigger, only the system admin user should be able to delete the task. you have to assign a task to the other user to make it visible in their org

```
trigger taskThatCanBeDeletedBySA on Task (before delete)
{
    Id pid = userinfo.getProfileId();
    Profile pname=[select Name from Profile where id=:pid];

    for(Task taskObj:Trigger.old){
        if(pname.Name != 'System Administrator'){
            taskObj.addError('No Access for Deletion');
        }
    }
}
```



Test Case:

1. Create a new user

(setup > home > quick find > users > new user) make sure the new user is not of system administrator profile and email id field is your working email id. we created a 'premium user' for this example.

	Alias	Username	Email	Profile
<input type="checkbox"/>	Admin	admin	admin@salesforce.com	System Administrator
<input type="checkbox"/>	Christie	christy01230000000000000000000000000000	christy01230000000000000000000000000000@salesforce.com	Customer Free User
<input type="checkbox"/>	Customer Test User	customer01230000000000000000000000000000	customer01230000000000000000000000000000@salesforce.com	Customer
<input type="checkbox"/>	Employee	employee01230000000000000000000000000000	employee01230000000000000000000000000000@salesforce.com	System Administrator
<input type="checkbox"/>	Manager	manager01230000000000000000000000000000	manager01230000000000000000000000000000@gmail.com	Manager
<input type="checkbox"/>	User Integration	integrator01230000000000000000000000000000	integrator01230000000000000000000000000000@salesforce.com	Analytics Cloud Integration User
<input type="checkbox"/>	user_premium	user_premium01230000000000000000000000000000	user_premium01230000000000000000000000000000@gmail.com	Empire.com - App Subsidiary User
<input type="checkbox"/>	User Security	usersec01230000000000000000000000000000	usersec01230000000000000000000000000000@salesforce.com	Analytics Cloud Security User

2. Now create a new task and assign (from your system admin org) it to the newly created user (premium user).

New Task

Task Information

Assigned To	* Status
premium user X	In Progress
1 Total Task	
* Subject	Name
Test Task	Search Contacts... X
Due Date	Related To
Calendar icon	Search Accounts... X
* Priority	
High	

Description Information

Comments

[Save & New](#) [Cancel](#) [Save](#)

Task **Test Task** Task **Test Task** was created. [X](#)

[Mark Complete](#) [Edit Comments](#) [Change Date](#) [Create Follow-Up Task](#) [▼](#)

Name	Related To

Details [Related](#)

Assigned To premium user	Status In Progress
Subject Test Task	Name
Due Date	Related To
Priority High	Last Modified By Sanjog Kawade, 7/6/2022, 3:55 PM
Created By Sanjog Kawade, 7/6/2022, 3:55 PM	Comments



3. Log in as the newly created user. You can view the assigned tasks in that org.

The screenshot shows the Salesforce Task Details page. The task is titled "Test Task" and is assigned to a user named "premium user". The task has a due date of "7/5/2022" and is marked as "In Progress". The "Details" tab is selected, showing the task's properties. The "Related" tab is also visible. On the left, there is a sidebar for "Overdue Tasks" which lists two items: "follow up" and "Test Task".

4. Try to delete the assigned task. The following error message is displayed. Since the user is not a system administrator, the premium user cannot delete the task.

The screenshot shows the same Salesforce Task Details page as before, but with a modal dialog box displayed over the task details. The dialog box contains the message "No Access for Deletion". This indicates that the user, who is a "premium user" and not a system administrator, does not have the necessary permissions to delete the task.

Example 8

Create a duplicate lead when a lead is inserted.

CheckRecursive (apex class)

```
public class checkRecursive {  
    private static boolean run = true;  
  
    public static boolean runOnce()  
    {  
        if(run)  
        {  
            run=false;  
            return true;  
        }  
        else  
        {  
            return run;  
        }  
    }  
}
```

DuplicateLead (apex trigger)

```
trigger duplicateLead on Lead (after insert) {  
  
    if(checkRecursive.runOnce()){  
  
        List<Lead> leadList = new List<Lead>();  leadList  
        = Trigger.new.deepClone();  
        insert leadList;  
    }  
}
```

Test Case:

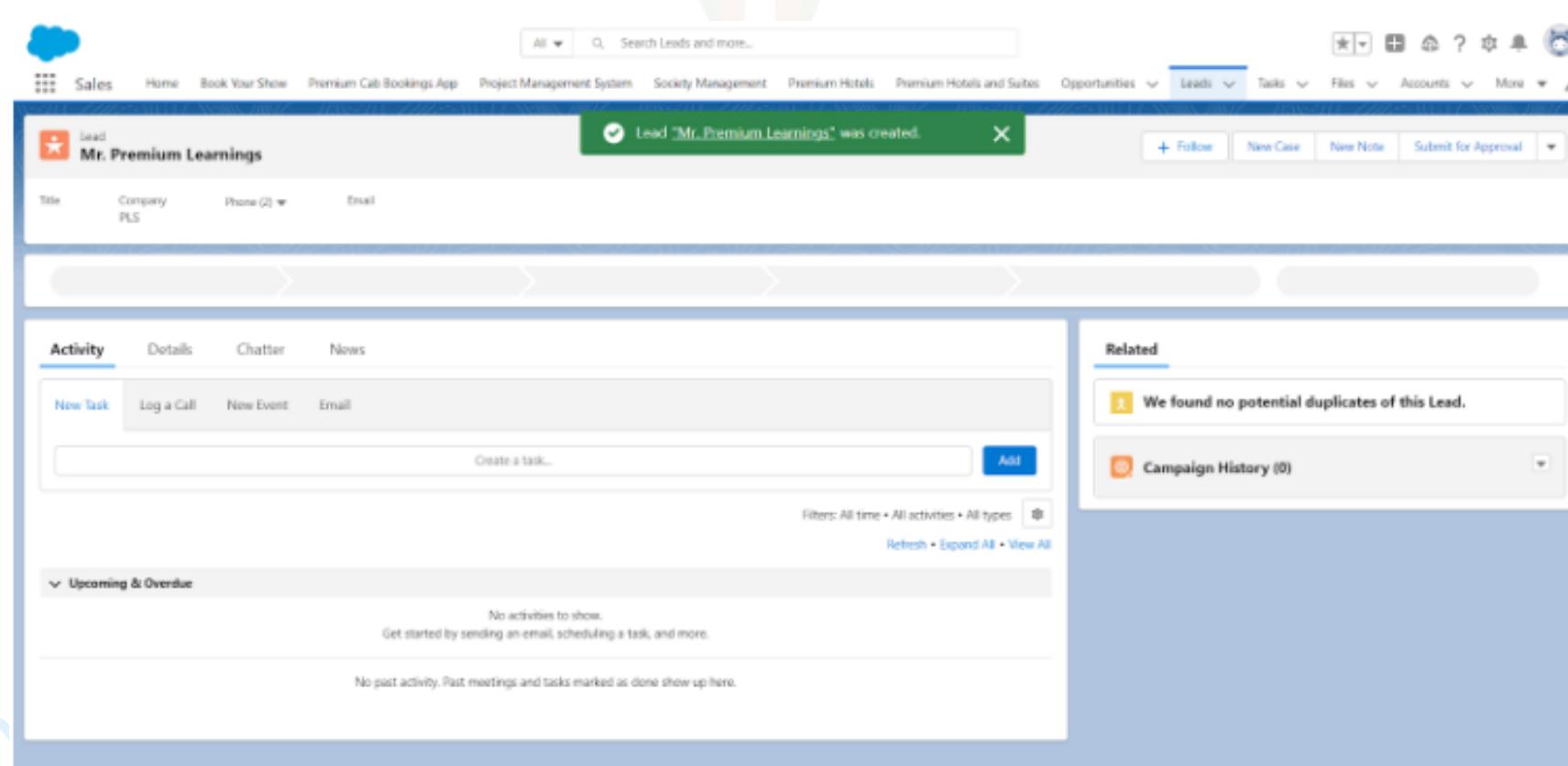
1. Create a new Lead. Click Save.

New Lead

Lead Information

Lead Owner  Sanjog Kawade	Phone
Salutation Mr.	Mobile
First Name Premium	Fax
*Last Name Learnings	Email
*Company PLS	Website
Title	*Lead Status Open - Not Contacted
Lead Source --None--	Rating
Industry --None--	
Annual Revenue	
<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>	

2. The record is created

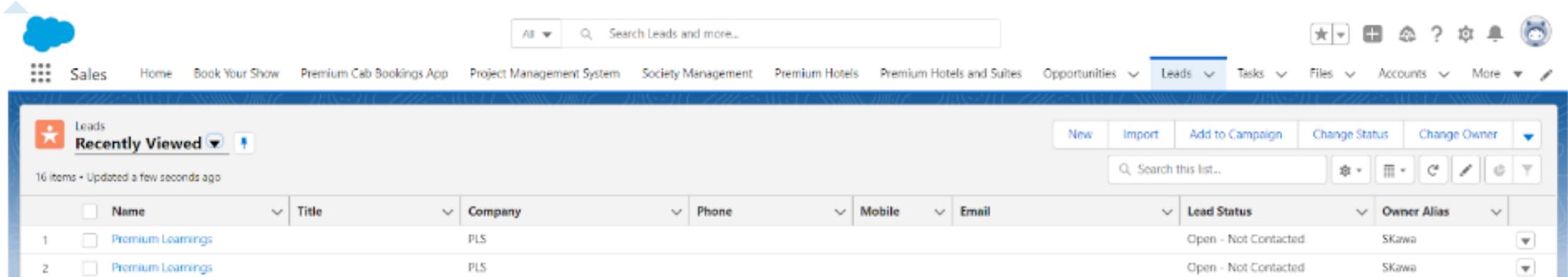


The screenshot shows the Salesforce Lead creation success page. At the top, a banner says "Lead 'Mr. Premium Learnings' was created." Below it, the lead record is displayed with the following details:

- Title:** PLS
- Company:** Premium Learnings
- Phone (G):** (Not shown)
- Email:** (Not shown)

The "Activity" section shows tabs for "New Task", "Log a Call", "New Event", and "Email". The "Related" section shows a message: "We found no potential duplicates of this Lead." and a link to "Campaign History (0)".

3. you can see a duplicate record with the same details is created.



The screenshot shows a CRM application's lead management screen. At the top, there are navigation links: Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, More, and a search bar labeled "Search Leads and more..". Below the header is a toolbar with buttons for New, Import, Add to Campaign, Change Status, and Change Owner, along with a search field and filter icons. The main area displays a table titled "Recently Viewed" with 16 items. The columns are: Name, Title, Company, Phone, Mobile, Email, Lead Status, and Owner Alias. Two entries are visible:

	Name	Title	Company	Phone	Mobile	Email	Lead Status	Owner Alias
1	Premium Learnings		PLS				Open - Not Contacted	SKawa
2	Premium Learnings		PLS				Open - Not Contacted	SKawa

Example 9

Upload any pdf file into the Document first. Send an email as an attachment to the lead email Id when a lead is created.

Email Subject: Welcome

Body: Please find the attached PDF.

On App Launcher search for Email Template then Create a new Email Template with Name=" Pdf Attached Email for Lead", Subject="Welcome", Body= "Please find the attached PDF." and Related Entity type= Lead and Save it.

Later go to its related tab and add the file which u want to attach to the mail.

Note: When u send single mail the actual file will be sent.

But if u send a list of emails then the files will be sent in the link format where if they click on that link then the file opens.

```
trigger sendEmailWithPdfFileAttached on Lead (after insert)
{
```

```
    List<Messaging.SingleEmailMessage> emailList= new
    List<Messaging.SingleEmailMessage>();
```

/*On App Launcher search for Email Template then Create a new

Email Template with Name="Pdf Attached Email for Lead", Subject="Welcome", Body= "Please find the attached PDF." and

Related Entity type= Lead and Save it.

Later go to its related tab and add the file which u want to attach to the mail.

Note : When u send single mail the actual file will be send.

But if u send a list of mails then the files will be send in the

link format where if they click on that link then the file opens.*/

```
EmailTemplate emailTemplate = [Select
    Id,Subject,Description,HtmlValue,DeveloperName,Body
  from EmailTemplate where name='Pdf Attached Email
  for Lead'];

for(Lead lObj:Trigger.new){
    if(lObj.Email!=null){
        Messaging.SingleEmailMessage mail = new
            Messaging.SingleEmailMessage();
        mail.setTargetObjectId(lObj.Id);
        mail.setSenderDisplayName('System Administrator');
        mail.setUseSignature(false);
        mail.setBccSender(false);
        mail.setSaveAsActivity(false);
        mail.setTemplateID(emailTemplate.Id);
        mail.toAddresses = new String[]{lObj.Email};
        emailList.add(mail);
    }
}

if(emailList.size()>0){
    Messaging.SendEmailResult[] results =
        Messaging.sendEmail(emailList);
    if (results[0].success)
    {
        System.debug('The email was sent successfully.');
    }
    else {
        System.debug('The email failed to send:
            '+ results[0].errors[0].message);
    }
}
```



Test Case:

1. Create a new lead. Make sure email id is populated

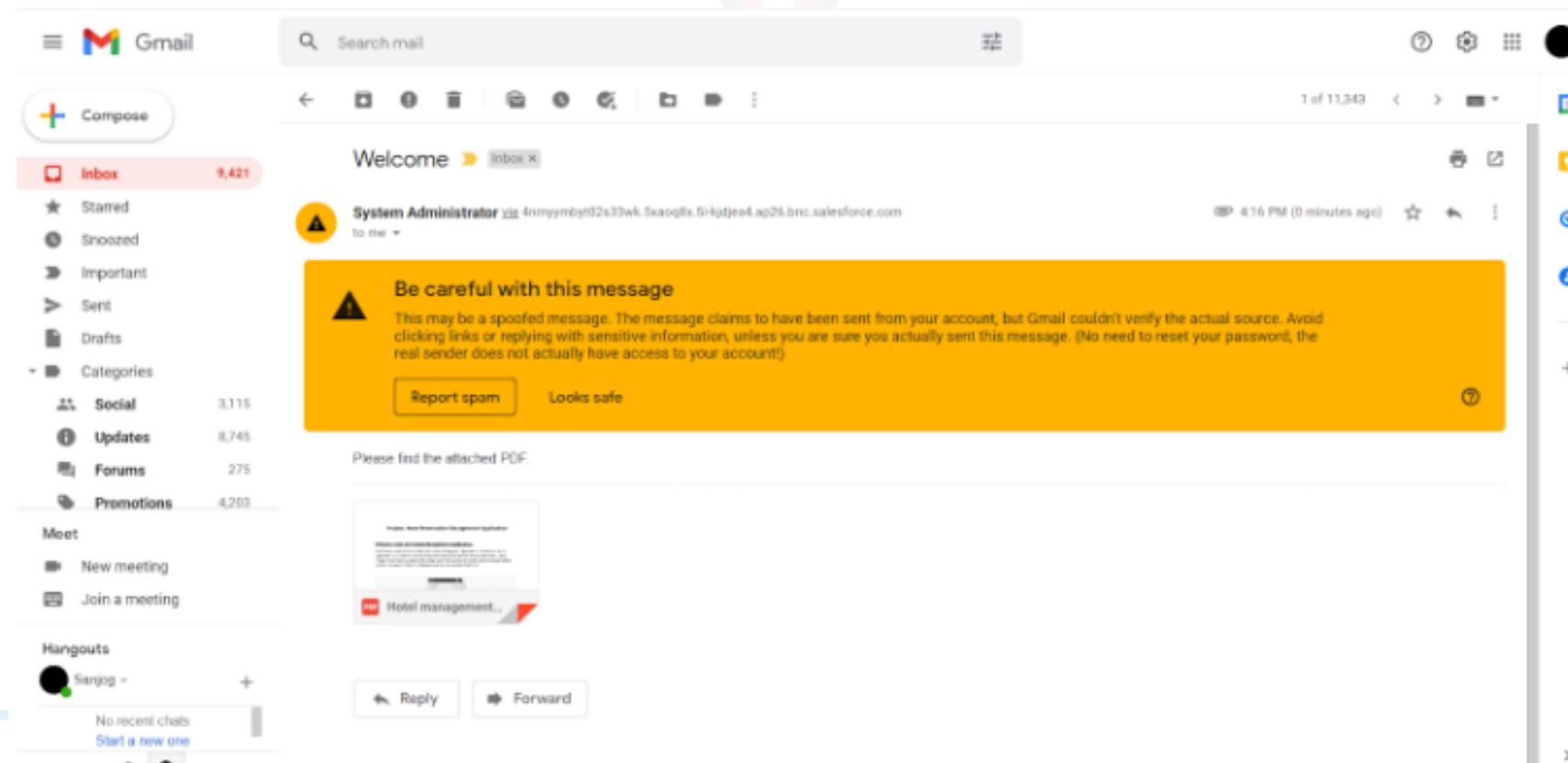
New Lead

Lead Information

Lead Owner  Sanjog Kawade	Phone <input type="text"/>
* Name <input type="text" value="Premium"/>	Mobile <input type="text"/>
Salutation <input type="text" value="Mr."/>	Fax <input type="text"/>
First Name <input type="text" value="Learnings"/>	Email <input type="text" value="kawadecsanjog@gmail.com"/>
* Last Name <input type="text" value="PLS"/>	Website <input type="text"/>
* Company <input type="text" value="Learnings"/>	* Lead Status <input type="text" value="Open - Not Contacted"/>
Title <input type="text"/>	Rating <input type="text"/>
Lead Source <input type="text" value="--None--"/>	Nr. of Employees <input type="text"/>
Industry <input type="text" value="--None--"/>	
Annual Revenue <input type="text"/>	

Buttons: Cancel | Save & New | Save

2. You will receive an email with attachment as shown



Example 10

Write a trigger on Contact, when contact is inserted an email should be sent to the contact email id with a specified template. So first you need to create a text template. The template design is below. Template name: Email Template to send After Contact is Inserted Subject: A new Contact was Created. Body: Dear “Contact Name”,

A new contact was created with Name: “Contact Name” on “Contact CreatedDate”. Contact your System Administrator if it was not created by you.

Thank You

Regards System Administrator.

```
trigger sendingMailAfterInsert on Contact (after insert) {

    List<Messaging.SingleEmailMessage> emailList= new
    List<Messaging.SingleEmailMessage>();
    EmailTemplate emailTemplate = [Select
        Id,Subject,Description,HtmlValue,DeveloperName,Body
        from EmailTemplate where name='Email Template to
        send After Contact is Inserted'];

    for(Contact conObj:Trigger.new){
        if (conObj.Email != null) {
            Messaging.SingleEmailMessage mail = new
                Messaging.SingleEmailMessage();
            mail.setTargetObjectId(conObj.Id);
            mail.setSenderDisplayName('System Administrator');
            mail.setUseSignature(false);
            mail.setBccSender(false);
            mail.setSaveAsActivity(false);
            mail.setTemplateID(emailTemplate.Id);
            mail.toAddresses = new String[]{conObj.Email};
            emailList.add(mail);
        }
    }
}
```

```
if(emailList.size()>0){  
    Messaging.SendEmailResult[] results =  
        Messaging.sendEmail(emailList);  
    if (results[0].success)  
    {  
        System.debug('The email was sent successfully.');//  
    }  
    else {  
        System.debug('The email failed to send: '+  
            results[0].errors[0].message);  
    }  
}
```





Test Case:

1. Create a new contact as shown below. Make sure email field is populated.

New Contact

Contact Information

Contact Owner  Sanjog Kawade	Phone
*Name	Home Phone
Salutation Mr.	
First Name Premium	Mobile
*Last Name Learnings	Other Phone
Account Name Search Accounts...	Fax
Title	Email
Department	<input type="text" value="sanjog.premium@gmail.com"/>
Birthdate	Assistant
Reports To Search Contacts...	
<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>	

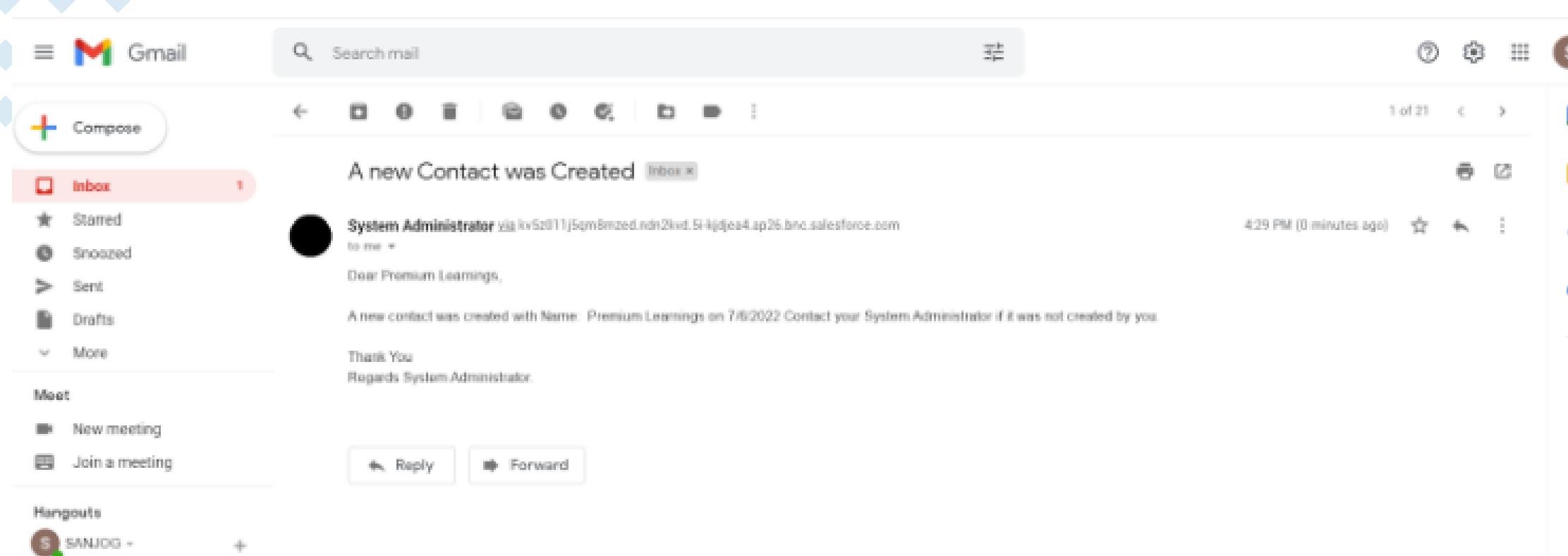
Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Contacts More

Contact Mr. Premium Learnings | A

Title	Account Name	Phone (0)	Email	Contact Owner
			sanjog.premium@gmail.com	 Sanjog Kawade
Related Details News <ul style="list-style-type: none"> We found no potential duplicates of this Contact. Opportunities (0) Cases (0) Campaign History (0) Notes & Attachments (0) 				
Activity Chatter <ul style="list-style-type: none"> New Task Log a Call New Event Email Create a task... Add Filter: All time + All activities + All types Refresh • Expand All • View All No activities to show. Get started by sending an email, scheduling a task, and more. No past activity. Past meetings and tasks marked as done show up here. 				

https://premiumlearnings-9b-dev-ed.lightning.force.com/lightning/v/Book_Your_Show

2. Email is sent to the email id as shown below





Example 11

When an opportunity line item is created an email should go to Opportunity

Account Client Contact — Client Contact would be a field on Account lookup to contact.

```
trigger sendEmailToContact on OpportunityLineItem (after
insert)
{
    Set<Id> oppId =new Set<Id>();
    String userName;
    Map<Id,Opportunity> oppIdConEmailMap = new
    Map<Id,Opportunity>();

    for(OpportunityLineItem oppProd:Trigger.new){
        oppId.add(oppProd.OpportunityId);
        userName=oppProd.CreatedBy.Name;
    }

    List<Opportunity> oppList=[Select Id,
                                Account.Client_Contact__r.Email,
                                Account.Client_Contact__r.Last
                                Name,Account.Client_Contact__c From
                                Opportunity where Id=:oppId];

    if(oppList.size()>0){
        for(Opportunity opp:oppList){
            oppIdConEmailMap.put(opp.Id,opp);
        }
    }

    List<Messaging.SingleEmailMessage> emailList= new
    List<Messaging.SingleEmailMessage>();
}
```

```

for(OpportunityLineItem oppProd:Trigger.new){

    if(oppIdConEmailMap.containsKey(oppProd.OpportunityId))
    {

        Opportunity oppObj =
                oppIdConEmailMap.get(oppProd.OpportunityId);
        String sendTo=oppObj.Account.Client_Contact__r.Email;

        if(sendTo!=null){

            Messaging.SingleEmailMessage mail = new
            Messaging.SingleEmailMessage();
            mail.setTargetObjectId(oppObj.Account.Client_Contact__c);
            mail.setSenderDisplayName('System Administrator');
            mail.setUseSignature(false);
            mail.setBccSender(false);
            mail.setSaveAsActivity(false);
            mail.setSubject('New Opportunity Product was Created.');
            String body = 'Dear' +
                    oppObj.Account.Client_Contact__r.LastName+', <br/>';
            body += 'Your Order has been proceed.<br/>';
            body += 'Order detail is below.<br/>';
            body += 'Product Name : '+oppProd.Name+'.<br/>';
            body += 'Product Code : '+oppProd.ProductCode+'.<br/>';
            body += 'Unit Price : '+oppProd.UnitPrice+'.<br/>';
            body += 'List Price : '+oppProd.ListPrice+'.<br/>';
            body += 'Thanks <br/>'+userName;
            mail.setHtmlBody(body);
            mail.toAddresses = new String[]{sendTo};
            emailList.add(mail);
        }
    }
}

```

```
if(emailList.size()>0){  
    Messaging.SendEmailResult[] results =  
        Messaging.sendEmail(emailList);  
    if (results[0].success)  
    {  
        System.debug('The email was sent successfully.');//  
    }  
    else {  
        System.debug('The email failed to send: '+  
            results[0].errors[0].message);  
    }  
}
```



Test Case:

1. Create contact and make sure the email field is populated

New Contact

Contact Information

Contact Owner  Sanjog Kawade	Phone
*Name	Home Phone
Salutation Mr.	Mobile
First Name Premium	Other Phone
*Last Name Learnings	Fax
Account Name Search Accounts...	Email sanjog.premium@gmail.com
Title	Assistant
Department	Reports To Search Contacts...
Birthdate	Cancel Save & New Save

2. Create a new account and make sure the client contact field is populated with the above created contact

Type --None--	Ownership --None--
Industry --None--	Employees
Annual Revenue	SIC Code
Premium <input type="checkbox"/>	
out of business <input type="checkbox"/>	
No of open opportunities	
RecordTypeid --None--	
Total Opportunity Amount	
Client Contact	
 Premium Learnings	X
Complete Opportunity Amount	
Sales Rep	Cancel Save & New Save



3. Now create a new opportunity and make sure the related account is the above-created account.

New Opportunity

Opportunity Information

Opportunity Owner Sanjog Kawade	Amount
Private <input type="checkbox"/>	* Close Date 7/20/2022
* Opportunity Name Premium Learnings	Next Step
Account Name Premium Learnings	* Stage Id. Decision Makers
Type --None--	Probability (%) 60%
Lead Source --None--	Primary Campaign Source Search Campaigns...
Approval status --None--	
Client Contact Search Contacts...	
Handoff Attached --None--	

Cancel Save & New Save

Active Customer project

Notice the 'products'(also called opportunity line item) in related lists of opportunity.

Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Contacts More

Opportunity Premium Learnings

Account Name: Premium Learnings Close Date: 7/20/2022 Amount: Opportunity Owner: Sanjog Kawade

Activity Details Chatter

New task: Log a Call, New Event, Email

No activities to show. Get started by sending an email, scheduling a task, and more.

Upcoming & Overage: No past activity. Past meetings and tasks marked as done show up here.

Related

- Products (0)
- Notes & Attachments (0)
- Contact Roles (0)
- Partners (0)
- Stage History (0)



a. Create a new product(click on arrow next to products as visible in above image and click add products) as shown below

Choose Price Book

Select a Price Book to associate with this opportunity. You can add products only from the Price Book you associate with this opportunity. Changing the Price Book deletes all products from the opportunity.

Price Book

Cancel
Save

Choose any product and click next

Add Products
Price Book: Standard

Search Products...	Show Selected (1)	Product Name	Product Code	List Price	Product Description	Product Family
		<input checked="" type="checkbox"/> GenWatt Diesel 100kW	GC1000	\$100,000.00		electronic
		<input type="checkbox"/> GenWatt Diesel 10kW	GC1020	\$5,000.00		
		<input type="checkbox"/> GenWatt Diesel 200kW	GC1040	\$25,000.00		
		<input type="checkbox"/> GenWatt Gasoline 2000kW	GC5060	\$150,000.00		
		<input type="checkbox"/> GenWatt Gasoline 300kW	GC5020	\$25,000.00		
		<input type="checkbox"/> GenWatt Gasoline 750kW	GC5040	\$75,000.00		
		<input type="checkbox"/> GenWatt Propane 100kW	GC3070	\$15,000.00		
		<input type="checkbox"/> GenWatt Propane 1500kW	GC3060	\$120,000.00		
		<input type="checkbox"/> GenWatt Propane 500kW	GC3040	\$50,000.00		
		<input type="checkbox"/> Installation: Industrial - High	IN7080	\$85,000.00		
		<input type="checkbox"/> Installation: Industrial - Low	IN7040	\$20,000.00		
		<input type="checkbox"/> Installation: Industrial - Medium	IN7060	\$50,000.00		
		<input type="checkbox"/> Installation: Portable	IN7020	\$5,000.00		
		<input type="checkbox"/> SLA: Bronze	SL9020	\$10,000.00		
		<input type="checkbox"/> SLA: Gold	SL9060	\$30,000.00		
		<input type="checkbox"/> SLA: Platinum	SL9080	\$40,000.00		
		<input type="checkbox"/> SLA: Silver	SL9040	\$20,000.00		

Cancel
Next

Fill in the fields and click save.

Edit Selected Products

*Product	*Quantity	*Sales Price	Date	Line Description
1 GenWatt Diesel 100kW	3.00	\$100,000.00	7/14/2022	

Back
Cancel
Save

Example 12

Create a fields on Opportunity as Client Contact, Once an Opportunity Client Contact updates then update Account Client Contact with same on Opportunity Client Contact.

```

trigger accClientConOnUpdateOfOppClientCon on Opportunity
(after update)
{
    Set<Id> oppId= new Set<Id>();

    for(Opportunity opp : Trigger.new) {

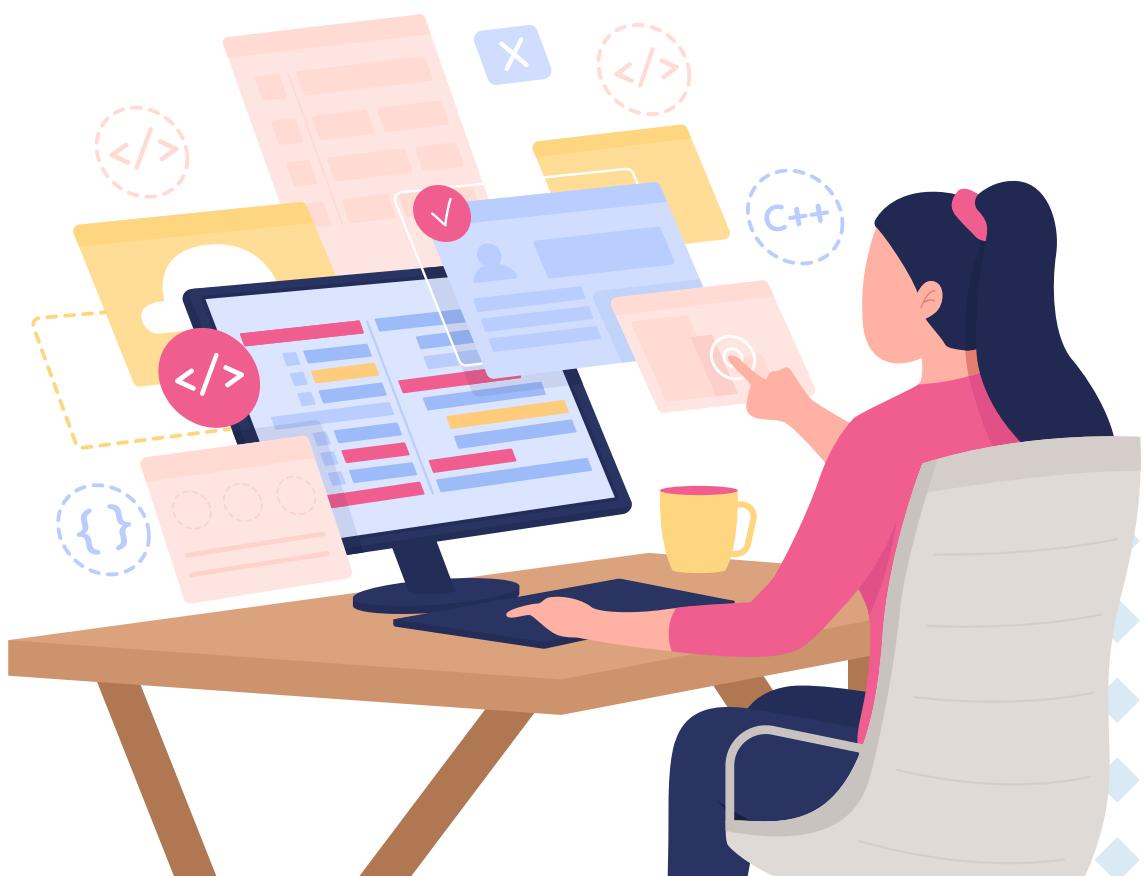
        //if the Client Contact of Opportunity is null then we
        //are not changing the Account Client Contact
        if(opp.Client_Contact__c!=null && opp.AccountId!=null)
        {
            if(opp.Client_Contact__c !=
                Trigger.oldMap.get(opp.Id).Client_Contact__c)
            {
                oppId.add(opp.Id);
            }
        }
    }

    List<Account> accList = new List<Account>();
    List<Opportunity> oppList = [Select Id,
        Client_Contact__c,AccountId,Account.Client_Contact__c
        from Opportunity where Id=:oppId];

    if(oppList.size()>0){
        for(Opportunity opp:oppList){

```

```
if(opp.Client_Contact__c !=  
    opp.Account.Client_Contact__c)  
{  
    Account accObj=new Account();  
    accObj.Id=opp.AccountId;  
    accObj.Client_Contact__c = opp.Client_Contact__c;  
    accList.add(accObj);  
}  
}  
}  
if(accList.size()>0){  
    update accList;  
}  
}
```





Test Case:

1. Create an account and keep the client contact custom field blank.

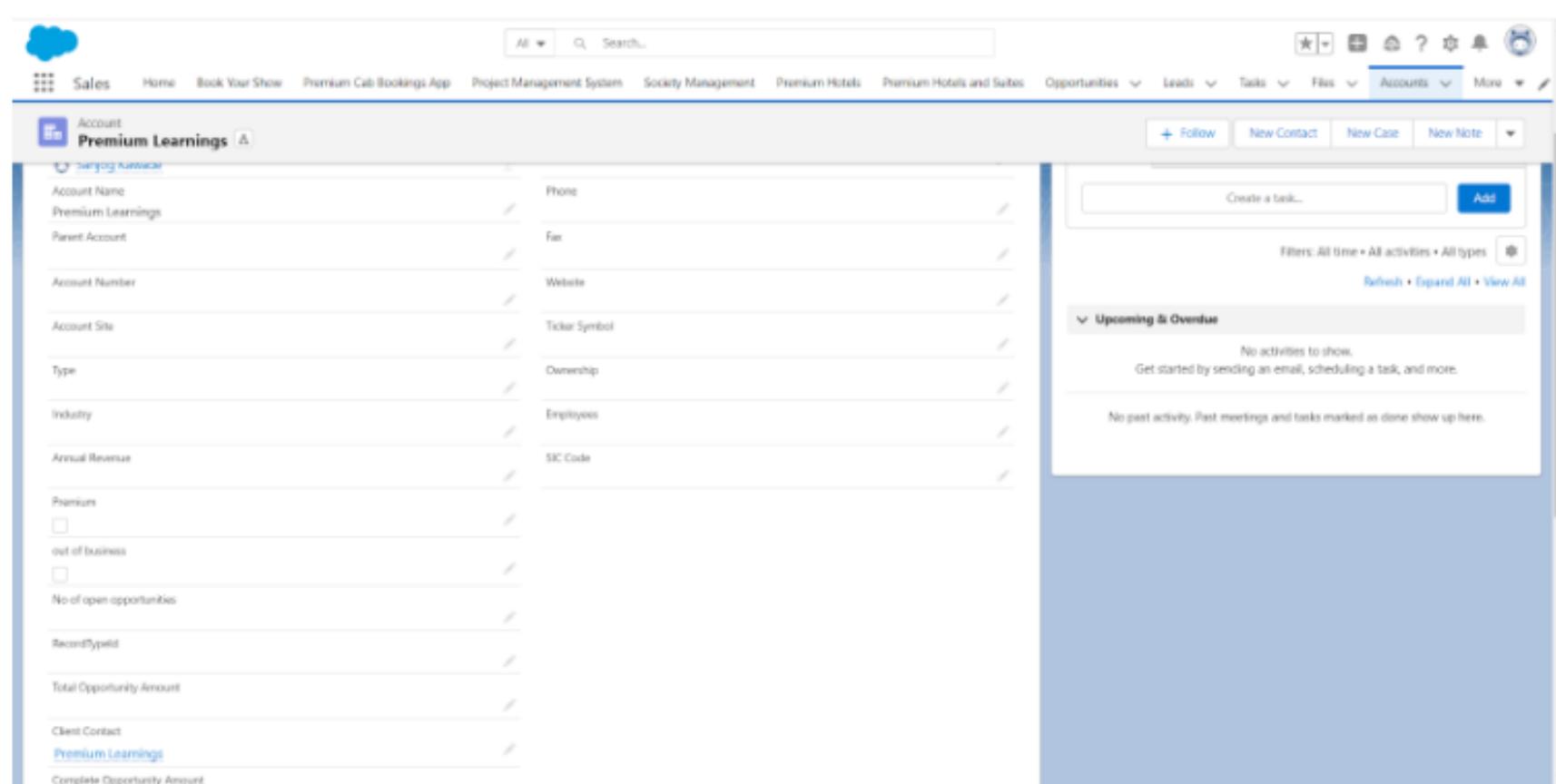
The screenshot shows the Salesforce Account page for 'Premium Learnings'. The 'Client Contact' field is empty. Other fields visible include Account Name, Phone, Email, Website, Type, Industry, Annual Revenue, Premium status, RecordTypeid, Total Opportunity Amount, and Client Contact.

2. Create an opportunity and save it. Now update the client contact field of the just created opportunity(select any contact) and select the above created account as the parent account.

The screenshot shows the Salesforce Opportunity page for 'Premium Learnings'. The 'Client Contact' field is populated with 'Premium Learnings'. Other fields visible include Account Name, Close Date, Amount (\$300,000.00), Opportunity Owner (Sanjog Kawade), Stage (Id. Decision Makers), and various tabs like Perception Analysis, Proposal/Price Quote, Negotiation/Review, and Closed.



3. Now go to the account record we created in step 1 and check the client contact field, it was blank earlier and is now populated with client contact field we selected in step 2, as shown in below image.



Example 13

Create an asset when create an OpportunityLineItem with associated Account. Create asset custom object with lookup relationship with account

```

trigger accountRelatedAsset on OpportunityLineItem
(after insert)
{
    Set<Id> oppId=new Set<Id>();

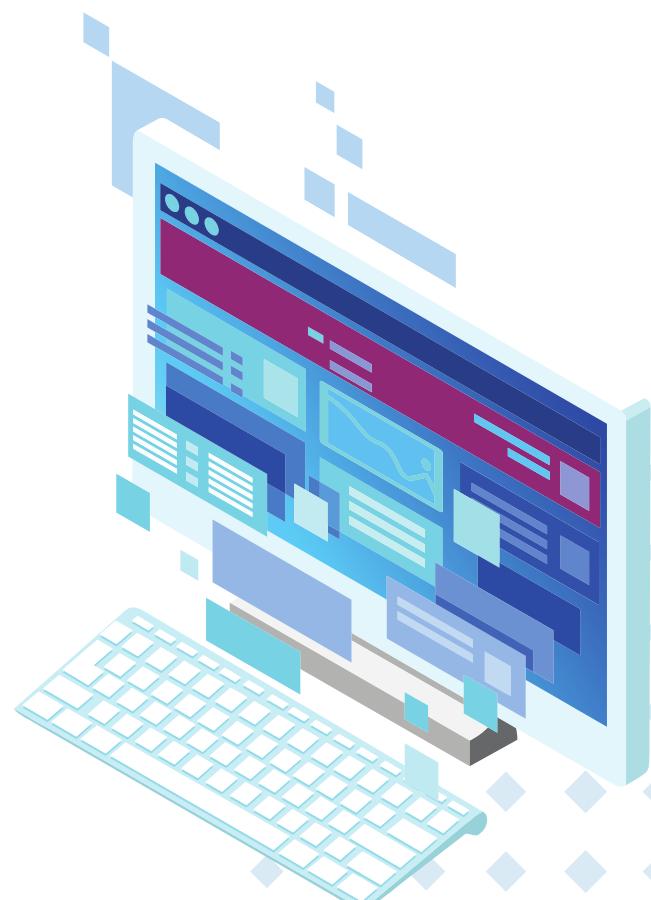
    for(OpportunityLineItem oppProd:Trigger.new){
        oppId.add(oppProd.OpportunityId);
    }

    List<Asset> assList=new List<Asset>();
    List<Opportunity> oppList =[Select Id,AccountId,
        Account.Name from Opportunity where Id=:oppId];

    if(oppList.size()>0){
        for(Opportunity opp:oppList){
            if(opp.AccountId!=null){
                Asset assObj=new Asset();
                assObj.Name=opp.Account.Name+
                    'OpportunityLineItem';
                assObj.AccountId=opp.AccountId;
                assList.add(assObj);
            }
        }
    }

    if(assList.size()>0){
        insert assList;
    }
}

```





Test Case:

1. create an opportunity with related account field populated as shown in below image

2. Now create a opportunity line item/opportunity product (product in the related list of opportunity, visible in the right of the above image) click on the arrow next to products > add products > price book - standard (if a window appears to select price book) > select any product as shown below. Click Next.

Product Name	Product Code	List Price	Product Description	Product Family
GenWatt Diesel 1000kW	GC1060	\$100,000.00		electronic
GenWatt Diesel 10kW	GC1020	\$5,000.00		
GenWatt Diesel 200kW	GC1040	\$25,000.00		
GenWatt Gasoline 2000kW	GC5060	\$150,000.00		
GenWatt Gasoline 300kW	GC5020	\$35,000.00		
GenWatt Gasoline 750kW	GC5040	\$75,000.00		
GenWatt Propane 100kW	GC3020	\$15,000.00		
GenWatt Propane 1500kW	GC3060	\$120,000.00		
GenWatt Propane 500kW	GC3040	\$50,000.00		
Installation: Industrial - High	IN7080	\$85,000.00		
Installation: Industrial - Low	IN7040	\$20,000.00		
Installation: Industrial - Medium	IN7060	\$50,000.00		
Installation: Portable	IN7020	\$5,000.00		
SLA: Bronze	SL9020	\$10,000.00		
SLA: Gold	SL9060	\$30,000.00		
SLA: Platinum	SL9080	\$40,000.00		
SLA: Silver	SL9040	\$20,000.00		

3. Add a desired quantity and date. click save

Edit Selected Products

Product	Quantity	Sales Price	Date	Line Description
1 GenMatt Diesel 1000kW	3.00	\$100,000.00	7/21/2022	

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

4. You can now see the product updated (in the right of below image) in the related list of opportunities.

All

Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Contacts More

Premium Learnings 1 Opportunity Product record was updated.

Follow New Case New Note Done

Activity Details Chatter

Opportunity Owner: Sanjog Kawade Amount: \$100,000.00 Close Date: 7/20/2022 Opportunity Owner: Sanjog Kawade

Decision Makers Perception Analysis Proposal/Price Quote Negotiation/Review Closed Mark Stage as Complete

Related

Products (1)

GenMatt Diesel 1000kW
Quantity: 3.00 Sales Price: \$100,000.00 Date: 7/21/2022

[View All](#)

Notes & Attachments (0)

[Upload Files](#) Or drop files



5. Asset record is now created as the apex trigger is fired.

The screenshot shows the Salesforce Asset detail page for an opportunity line item. The page has a header with a cloud icon, a search bar, and various navigation links. Below the header, there are tabs for 'Asset' and 'Premium Learnings OpportunityLineItem'. The main content area is divided into two sections: 'Details' and 'Activity'.

Details Section:

- Account:** Premium Learnings
- Contact:** [empty]
- Quantity:** [empty]
- Asset Name:** Premium Learnings OpportunityLineItem
- Product:** [empty]
- Product Code:** [empty]
- Serial Number:** [empty]
- Install Date:** [empty]
- Status:** [empty]
- Quantity:** [empty]
- Price:** [empty]
- Created By:** Sanjog Kawade, 7/6/2023, 5:56 PM
- Description:** [empty]
- Last Modified By:** Sanjog Kawade, 7/6/2023, 5:56 PM

Activity Section:

- Activity Types:** New Event, New Task, Log a Call, Email
- Event Input:** Set up an event...
- Filters:** All time • All activities • All types
- Upcoming & Overage:** No activities to show. Get started by sending an email, scheduling a task, and more.
- Past Activity:** No past activity. Past meetings and tasks marked as done show up here.

Example 14

Once Opportunity Line Item is added to Opportunity with the specified product, then Product Total Quantities must be deduct from the Product Object

e.g. If we have Total Quantity 100 on Product object and we have added 50 in line items , then product must be update $100-50 = 50$
 Create Total Quantity, Available Quantity Fields on Product2 object.

```
trigger updateProductTotalQuantities on OpportunityLineItem
(after insert)
{
    Set<Id> proId = new Set<Id>();

    for(OpportunityLineItem oppProd:Trigger.new){
        proId.add(oppProd.Product2Id);
    }

    Map<Id,Double> proIdQuantitySum =new Map<Id,Double>();
    List<AggregateResult> results=[select Product2Id,
                                    sum(Quantity)quantity from OpportunityLineItem
                                    where Product2Id in :proId group by Product2Id];

    if(results.size()>0){
        for(AggregateResult a: results){
            Id prodId = (Id)a.get('Product2Id');
            double TotalQuantity = (double)a.get('quantity');

            //Adding ProductId and sum(Quantity) to Map
            proIdQuantitySum.put(prodId,TotalQuantity);
        }
    }
}
```

```

List<Product2> proListToUpdate = new List<Product2>();
List<Product2> proList=[Select Id,Total_Quantities__c
from Product2 where Id=:proId];

if(proList.size()>0){
    for(Product2 pro : proList){

        //Checking if the ProductId is present in Map
        if(proIdQuantitySum.containsKey(pro.Id)){
            if(pro.Total_Quantities__c!=null){

                //Subtracting the Quantity
                pro.Total_Quantities__c =
                    pro.Total_Quantities__c-
                    proIdQuantitySum.get(pro.Id);
                proListToUpdate.add(pro);
            }
        }
    }
}

if(proListToUpdate.size()>0){
    update proListToUpdate;
}
}

```



Test Case:

- Select any record from products object and enter the Total Quantity field with value '50'.

	Product Name	Product Code	Product Description	Product Family
1	GenWatt Diesel 1000kW	GC1060		
2	GenWatt Diesel 10kW	GC1020		electronic
3	GenWatt Diesel 200kW	GC1040		

- We use GenWatt Diesel 1000kW for testing. Click on the arrow next to its name and click edit.

	Product Name	Product Code	Product Description	Product Family
1	GenWatt Diesel 1000kW	GC1060		
2	GenWatt Diesel 10kW	GC1020		electronic
3	GenWatt Diesel 200kW	GC1040		
4	GenWatt Gasoline 2000kW	GC3060		

3. Add total quantities as 50

Edit GenWatt Diesel 1000kW

*Product Name

Active

Product Code

Product Family

Total Quantities

Available Quantity

Created By

Sanjog Kawade, 4/16/2022, 2:01 PM

Last Modified By

Sanjog Kawade, 4/16/2022, 2:01 PM

Product Description

4. select any opportunity (or create one).



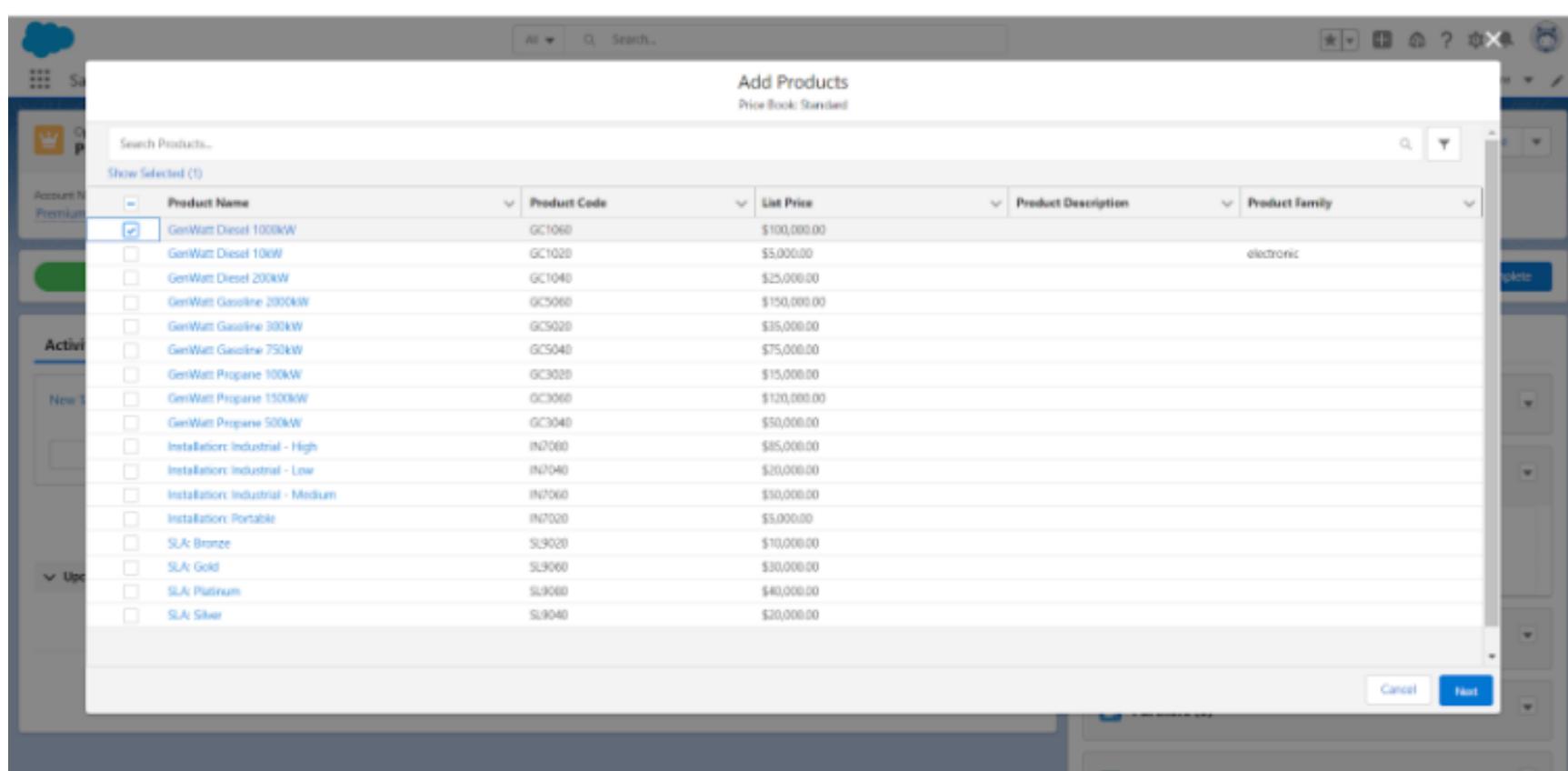
The screenshot shows the Salesforce Opportunities tab selected. The top navigation bar includes Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, File, Accounts, and More. Below the navigation is a search bar and a 'Recently Viewed' list titled 'Opportunities'. The list displays three opportunities: 'Premium Learnings', 'Edge Emergency Generator', and another 'Edge Emergency Generator' entry, all associated with 'Edge Communications' and 'Closed Lost' status.

Opportunity Name	Account Name	Account Site	Stage	Close Date	Opportunity Owner Alias
Premium Learnings	Premium Learnings		Not Decision Makers	7/6/2022	SKteam
Edge Emergency Generator	Edge Communications		Closed Lost	7/6/2022	SKteam
Edge Emergency Generator	Edge Communications		Closed Lost	7/6/2022	SKteam

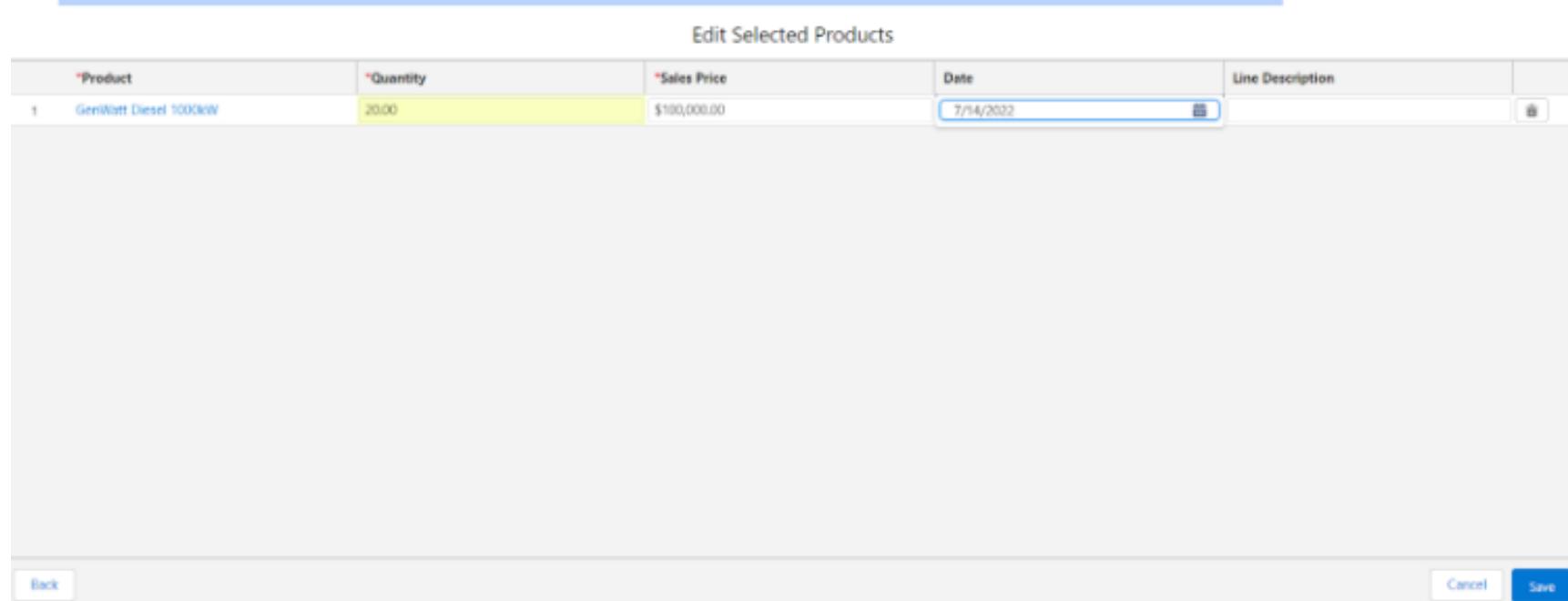
5. Add a product to that opportunity(you can see products in the opportunity related lists)

The screenshot shows a CRM interface for an opportunity named "Premium Learnings". The top navigation bar includes links for Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, and More. The main content area displays the opportunity details: Account Name (Premium Learnings), Close Date (2/20/2022), Amount (\$80.00), and Opportunity Owner (Surjeet Kaurde). Below this is a green progress bar with several steps, one of which is labeled "M. Decision Makers". To the right of the progress bar are buttons for Follow, New Case, New Note, and Done. Further down, there's a section for activities with buttons for New Task, Log a Call, New Event, and Email. A related section on the right lists Products (0), Notes & Attachments (0), Contact Roles (0), and Partners (0).

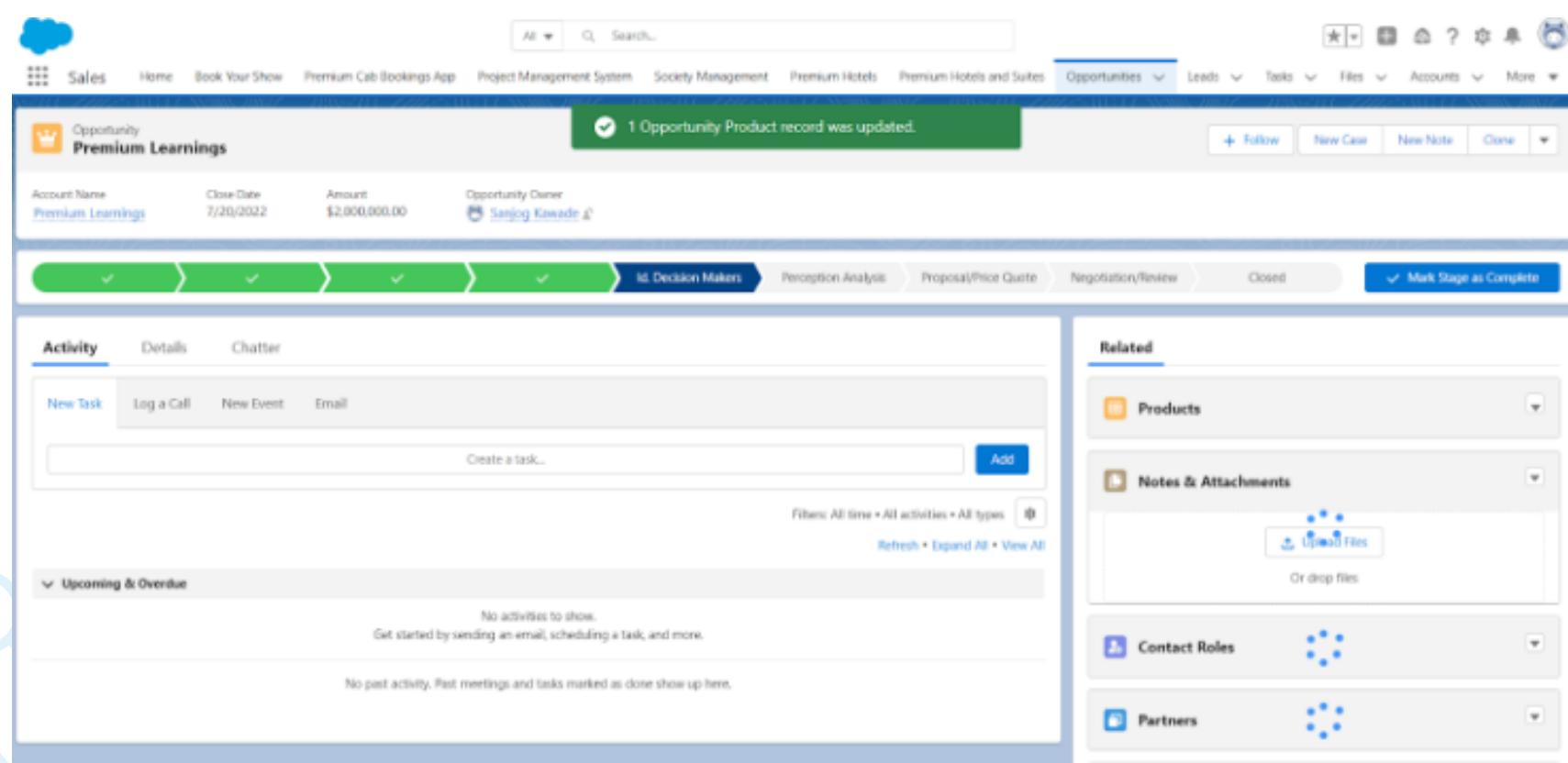
6. Select GenWatt Diesel 1000Kw and click next



7. set quantity as 20 and set any date in field. Click save



8. Opportunity product is created





9. if we check the product details of genWatt diesel 1000kW product, we can see the total quantities field updated now to 30.(you may need to refresh the page)

Product Name	Active
GenWatt Diesel 1000kW	<input checked="" type="checkbox"/>
Product Code	Product Family
GC1000	
Total Quantity	30
Available Quantity	

Created By: [Sengig Kawade](#), 4/16/2022, 2:01 PM Last Modified By: [Sengig Kawade](#), 7/7/2022, 3:39 PM

Example 15

The following Trigger will fires when we try to create the account with the same name i.e. Preventing the users to create Duplicate Accounts

```
trigger AccountDuplicateTrigger on Account (before insert,  
before update){  
  
    for(Account a:Trigger.new)  
    {  
  
        List<Account> acc=[Select id, Name from Account where  
        Name=:a.Name and Rating=:a.Rating];  
  
        if(acc.size()>0)  
        {  
            a.Name.addError('You Cannot Create the Duplicate  
            Account');  
        }  
    }  
}
```





Test Case:

1. Create an account with name Premium Learnings and save it

2. Create another account with same name and try to save it. You will get the following error on screen.



Example 16

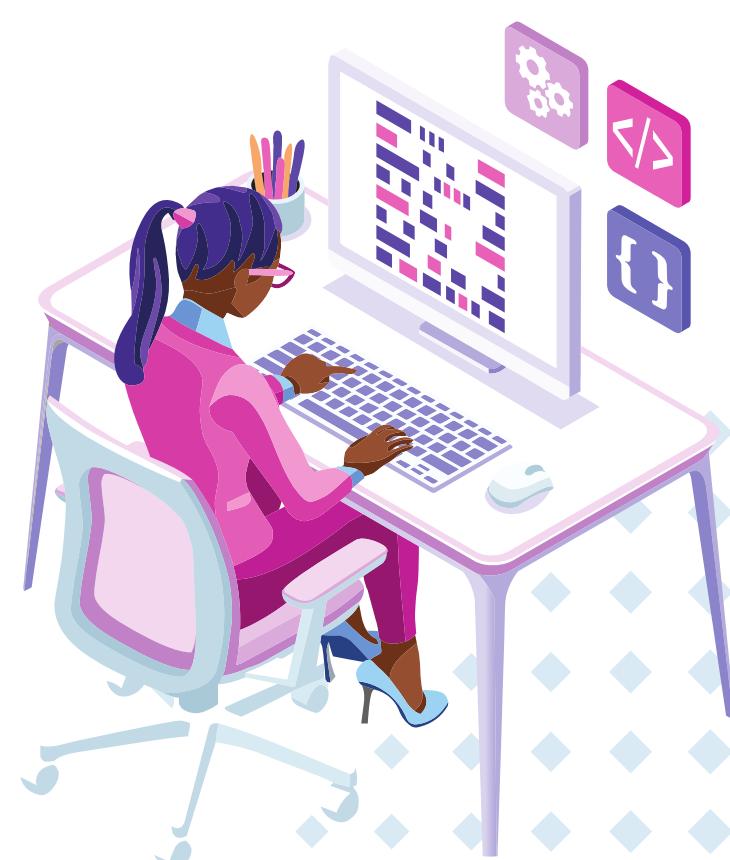
Create the object called “Books” and create the field “Price”(data type is Currency) under this object. Whenever we enter some amount of money in the Price field and once we click on save button, the value we entered in the Price field is 10% less than the actual price. This is applicable for while both inserting and updating records.

DiscountTrigger (apex trigger)

```
trigger DiscountTrigger on Book__c (before insert,
before update)
{
    List<Book__c> books = Trigger.new;
    PriceDiscount.applyDiscount(books);
}
```

PriceDiscount (apex class)

```
public class PriceDiscount {
    public static void applyDiscount(List<Book__c> books)
    {
        for (Book__c b :books){
            b.Price__c *= 0.9;
        }
    }
}
```





Test Case:

1. Create a new 'book' record as shown below and click save

The screenshot shows a 'New Book' dialog box. In the 'Information' section, the 'Book Name' field contains 'Premium Learnings' and the 'Price' field contains '\$10'. The 'Owner' field is set to 'Sanjog Kawade'. At the bottom of the dialog, there are three buttons: 'Cancel', 'Save & New', and 'Save'. The background of the dialog is light gray.

2. After the record is saved, the price we entered above(100) gets changed to 90

The screenshot shows the details page for the 'Premium Learnings' book record. The 'Details' tab is selected. The record shows a 'Book Name' of 'Premium Learnings', a 'Price' of '\$90', and was 'Created By' 'Sanjog Kawade' on '7/7/2022, 4:14 PM'. The 'Owner' is listed as 'Sanjog Kawade'. In the 'Activity' section, there is a button to 'Set up an event...' and a link to 'Upcoming & Overdue' activities. The background of the page is white.

Example 17

The following trigger will prevent the users from deleting the Accounts. This is because System Administrator has all the permissions, we cannot change the permissions.

```
trigger AccountDelete on Account (before delete)
{
    for(Account Acc:trigger.old)
    {
        acc.adderror('You Cannot Delete the Account Record');
    }
}
```





Test Case:

1. Open any account record(or create one) and try to delete it

The screenshot shows the Salesforce interface for managing accounts. The main area displays the details of an account named 'Premium Learnings'. On the right side, there's a sidebar titled 'Activity' which includes a 'Chatter' section and a 'Upcoming & Overdue' section. A context menu is open in the top right corner, listing various actions such as 'New Opportunity', 'Change Owner', 'Check for New Data', 'Submit for Approval', 'Edit', 'Delete', 'Sharing', and 'View Account History'. The 'Delete' option is clearly highlighted with a blue background.

2. The following error message will be displayed

This screenshot shows the same Salesforce account page as the previous one, but now with a modal dialog box overlaid. The dialog box has a white background and a dark border, containing the text 'You Cannot Delete the Account Record' in a standard font. This indicates that the deletion attempt was unsuccessful.

Example 18

Create a Custom field called “Number of Locations” on the Account Object (Data Type=Number)The following trigger creates the number of contacts that are equal to the number which we will enter in the Number of Locations field on the Account Object

```
trigger ContactsCreation on Account (after insert) {

    list<contact> listContact = new list<contact>();
    map<id,decimal> mapAcc=new map<id,decimal>();

    for(Account acc:trigger.new){
        mapAcc.put(acc.id,acc.Number_of_Locations__c);
    }
    if(mapAcc.size()>0 && mapAcc!=null){
        for(Id accId:mapAcc.keySet()){
            for(integer i=0;i<mapAcc.get(accId);i++){
                contact newContact=new contact();
                newContact.accountid=accId;
                newContact.lastname='contact'+i;
                listContact.add(newContact);
            }
        }
    }
    if(listContact.size()>0 && listContact!=null)
        insert listContact;
}
```

Test Case:

1. Create an account record. And input the number of locations field with a number(for example 3)

New Account

Account Information

Account Owner  Sanjog Kawade	Rating --None--
* Account Name Premium Learnings	Phone
Parent Account Search Accounts...	Fax
Account Number	Website
Account Site	Ticker Symbol
Type --None--	Ownership --None--
Industry --None--	Employees
Annual Revenue	SIC Code
Premium	

out of business [Cancel](#) [Save & New](#) [Save](#)

Click save

Billing City	Billing State/Province	Shipping City	Shipping State/Province
Billing Zip/Postal Code	Billing Country	Shipping Zip/Postal Code	Shipping Country

Additional Information

Customer Priority --None--	SLA --None--
SLA Expiration Date	SLA Serial Number
Number of Locations 3	Upsell Opportunity --None--
Active --None--	

Description Information

Description

[Cancel](#) [Save & New](#) [Save](#)



The screenshot shows the Dynamics 365 Accounts screen. At the top, there's a navigation bar with links like Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Risks, Accounts (which is selected), and More. A search bar at the top right says "Search Accounts and more...". Below the navigation is a green banner with the message "Account 'Premium Learnings' was created." On the main card, the account name is "Premium Learnings", and the account owner is "Sanjog Kawade". There are tabs for Type, Phone, Website, Account Owner, Account Site, and Industry.

2. we can see three contacts created automatically in related lists as shown below

This screenshot shows the "Related" section of the "Premium Learnings" account page. It includes sections for Contacts, Opportunities, Cases, and Notes & Attachments. The Contacts section displays three entries: "contact", "contact1", and "contact2", each with fields for Title, Email, and Phone. The Activity section shows a "New Task" button and a "Create a task..." input field. It also includes filters, a refresh button, and a "View All" link. The "Upcoming & Overage" section indicates "No activity to show" and "Get started by sending an email, scheduling a task, and more." The "Past activity" section indicates "No past activity. Past meetings and tasks marked as done show up here."



Example 19

The logic is when we create Customer Project for an Opportunity with the Status=Active, then Active Customer project check box on the Opportunity Detail page is automatically checked.

Create the object called “Customer Project” and create a Status field under this object with the picklist data type (Values=Active, Inactive). Create the relationship between this custom object and Opportunity so that the Customer Project is a related list to the Opportunity. Create Active Customer project - field on Opportunity object (Data Type=Checkbox)

```
trigger UpdateCPActivenOnOppty on Customer_Project__c
(after insert){
```

```
List<Opportunity> opps=new List<Opportunity>();
for(Customer_Project__c cp:Trigger.New){
    if(cp.Status__c=='Active'){
        Opportunity opp=new Opportunity(id=cp.Opportunity__c);
        opp.Active_Customer_Project__c = True;
        opps.add(opp);
    }
}
update opps;
```





Test Case:

1. Open any opportunity record(or create one) and make sure the Active Customer Project Checkbox is Unchecked.

Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Accounts More

All Search Opportunities and more...

Premium Learnings

Private Opportunity Name Premium Learnings Account Name Premium Learnings Type Lead Source Approval status Client Contact Premium Learnings Handoff Attached Active Customer project Product type Order Number Current Generator(s) Tracking Number Created By

Expected Revenue \$1,200,000.00 Close Date 7/20/2022 Next Step Stage Id. Decision Makers Probability (%) 60% Primary Campaign Source Main Competitor(s) Delivery/Installation Status Last Modified By

GenWatt Diesel 1000kW Quantity: 200 Sales Price: \$100,000.00 Date: 7/14/2022

View All

Notes & Attachments (0) Upload Files Or drop files

Contact Roles (0)

Partners (0)

Stage History (3+)

Stage: Id. Decision Makers Amount: \$1,200,000.00 Probability (%): 60% Expected Revenue: \$1,200,000.00

2. scroll down in the related lists to find 'customer project' related object

Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Accounts More

All Search Opportunities and more...

Premium Learnings

Account Name Premium Learnings Close Date 7/20/2022 Amount \$2,000,000.00 Opportunity Owner Sanjog Kawade

Id. Decision Makers Perception Analysis Propose/Price Quote Negotiation/Review Closed Mark Stage as Complete

Activity Details Chatter

New Task Log a Call New Event Email Create a task... Add Filter: All time • All activities • All types Refresh • Expand All • View All

Upcoming & Overdue No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

Related

Products (1)

GenWatt Diesel 1000kW Quantity: 200 Sales Price: \$100,000.00 Date: 7/14/2022

View All

Notes & Attachments (0) Upload Files Or drop files



The screenshot shows the Salesforce Opportunities page for an opportunity named 'Premium Learnings'. The main panel displays the opportunity details, including Expected Revenue (\$60,000.00), Close Date (7/30/2022), and Stage (Id. Decision Makers). Below the main panel, there are five related lists: Quotes (0), Approval History (0), Top X Designations (0), Customer Projects (0), and Stakeholders (0). A URL at the bottom of the page is: https://premiumlearnings-9b-dev-ed.lightning.force.com/lightning/r/000/0000000000000000000000000000000/relation/Customer_Projects__r/view.

3. Create a new customer project from related lists as shown below

The screenshot shows the same Salesforce Opportunities page for 'Premium Learnings'. In the Customer Projects (0) related list, a 'New' button is visible, indicating that a new customer project is being created. The rest of the interface and data points are identical to the previous screenshot.



4. Create a new Customer Project record and save it. (make sure Status field is active)

New Customer Project

Information

*Customer Project Name	Owner
Premium Learnings	Sanjog Kawade
Status	
Active	
Opportunity	
Premium Learnings	X

Cancel Save & New Save

5. Now check the opportunity record again to see the active customer project checklist is now checked.

Opportunity Premium Learnings

Activity Details Chatter

Opportunity Owner	Amount
Sanjog Kawade	\$2,000,000.00
Private	Expected Revenue
<input type="checkbox"/>	\$1,200,000.00
Opportunity Name	Close Date
Premium Learnings	7/20/2022
Account Name	Next Step
Premium Learnings	
Type	Stage
	Id. Decision Makers
Lead Source	Probability (%)
	60%
Approval status	Primary Campaign Source
Client Contact	
Premium Learnings	
Has Draft Attached	
Active Customer project	
<input checked="" type="checkbox"/>	
Product type	
Order Number	Main Competitor(s)

Related

- Products (1)

GenWatt Diesel 1000kW
Quantity: 20.00
Sales Price: \$100,000.00
Date: 7/14/2022
- Notes & Attachments (0)
- Contact Roles (0)
- Partners (0)
- Stage History (3+)

Example 20

Create “Sales Rep” field with the data type (Text) on the Account Object. When we create the Account record, the Account Owner will be automatically added to Sales Rep field. When we update the Account owner of the record, then also

```
trigger UpdateSalesRep on Account (Before insert,
Before Update)
```

```
{  
  
Set<Id>setAccOwner=new Set<Id>();  
for(Account Acc: trigger.new)  
{  
    setAccOwner.add(Acc.OwnerId);  
}}
```

```
Map<Id,User>User_map = new Map<Id,User>([select Name  
from User where id in:setAccOwner]);  
  
for(Account Acc: Trigger.new)  
{  
    User usr=User_map.get(Acc.OwnerId);  
    Acc.Sales_Rep__c=usr.Name;  
}  
}
```



Test Case:

1. Create a new account with only name field populated as shown below

New Account

Account Information

Account Owner Sanjog Kawade	Rating --None--
* Account Name Premium Learnings	Phone
Parent Account Search Accounts...	Fax
Account Number	Website
Account Site	Ticker Symbol
Type --None--	Ownership --None--
Industry --None--	Employees
Annual Revenue	SIC Code
Premium <input type="checkbox"/> out of business <input type="checkbox"/>	<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>

https://premiumlearnings-9b-dev-ed.lightning.force.com/lightning/l/Account/001500000NBn7EAA/view

Cloud Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Accounts More

Account Premium Learnings Account "Premium Learnings" was created.

Type Phone Website Account Owner Sanjog Kawade Account Site Industry

Related Details News

- We found no potential duplicates of this Account.
- Contacts (0)
- Opportunities (0)
- Cases (0)
- Notes & Attachments (0) Or drop files
- Partners (0)

Activity Chatter

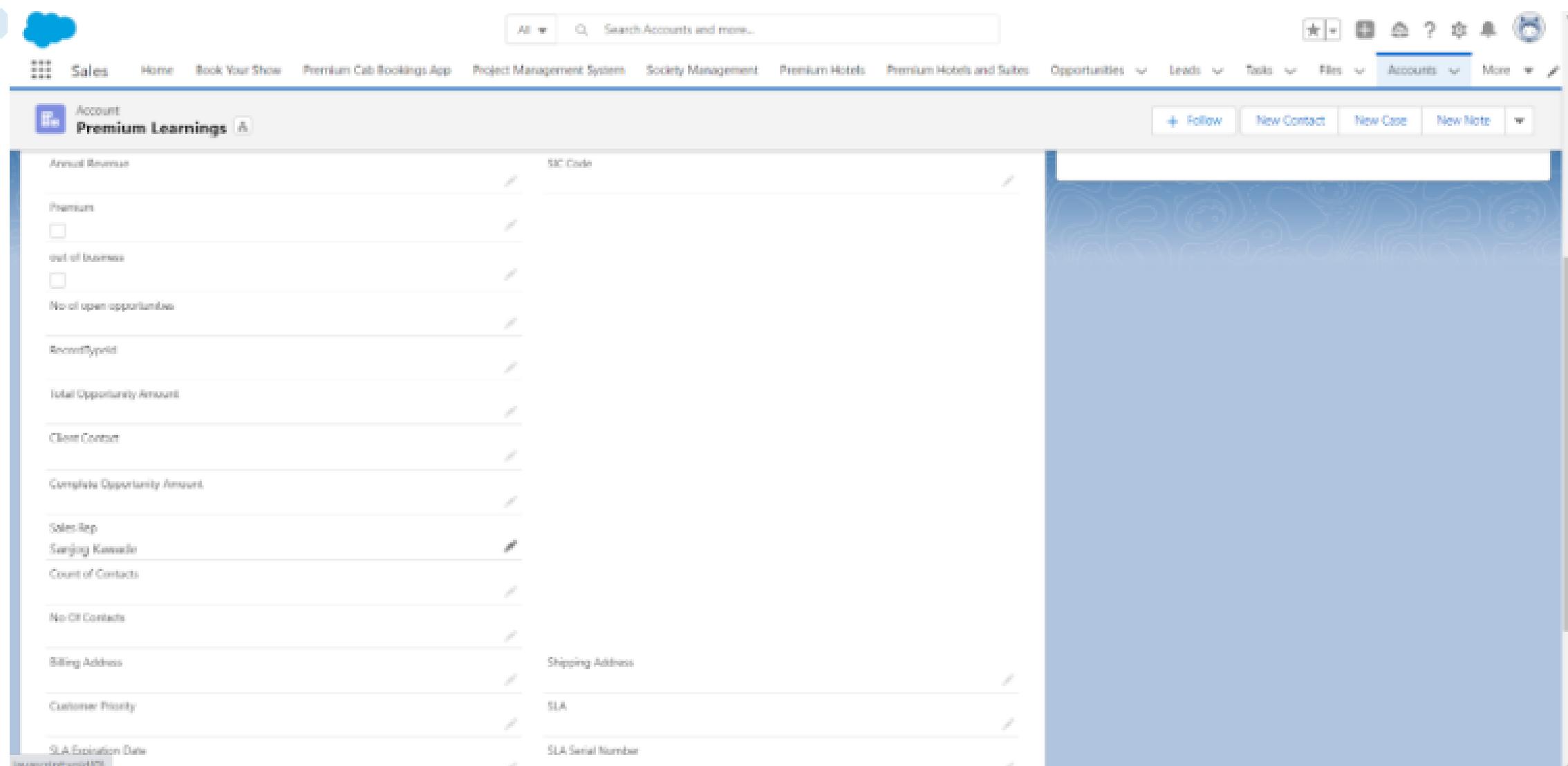
New Link Log a Call New Event Email Create a link... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

Upcoming & Ongoing No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

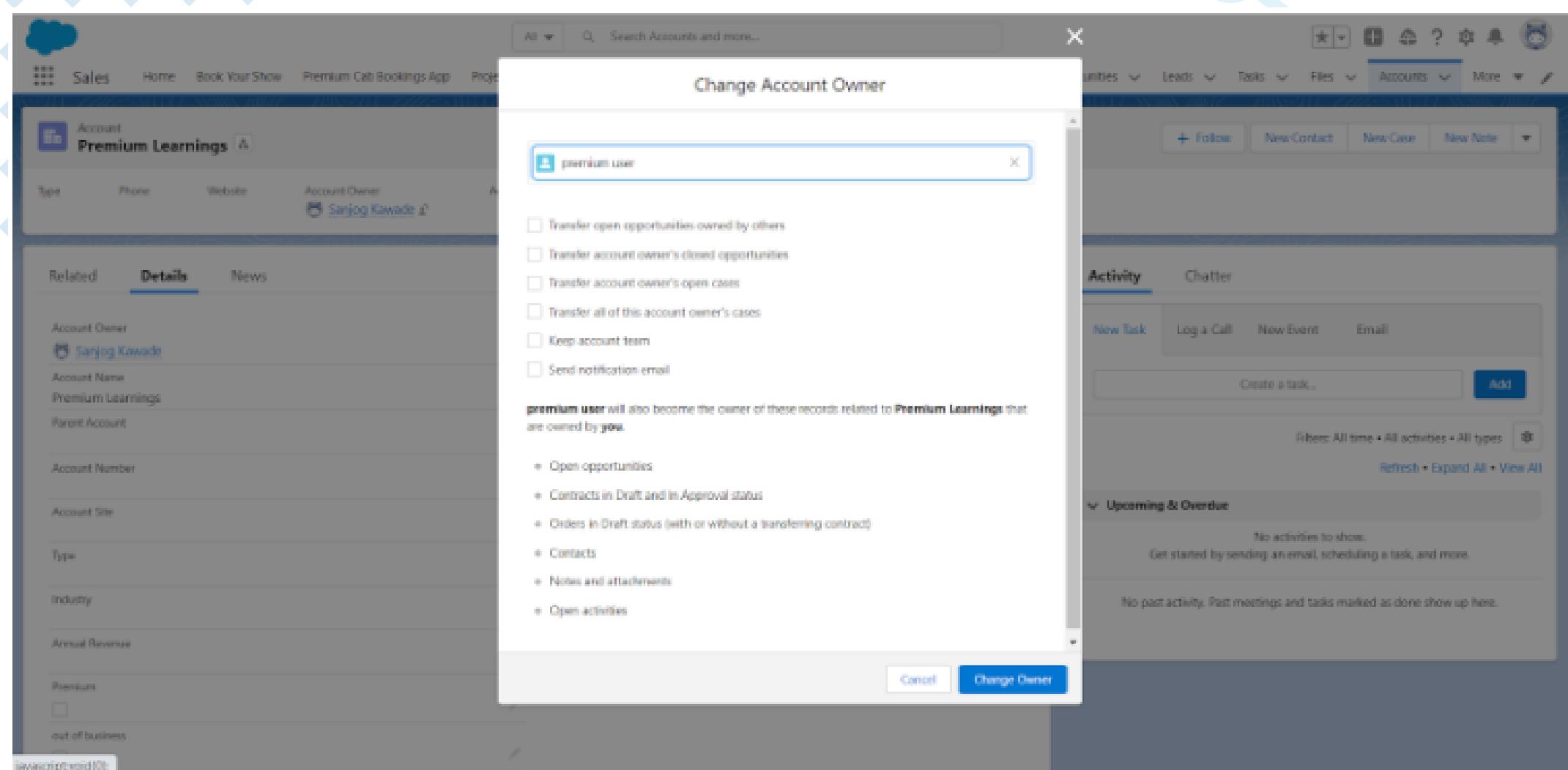
2. Now check the sales rep custom field and you will see your name populated in it.



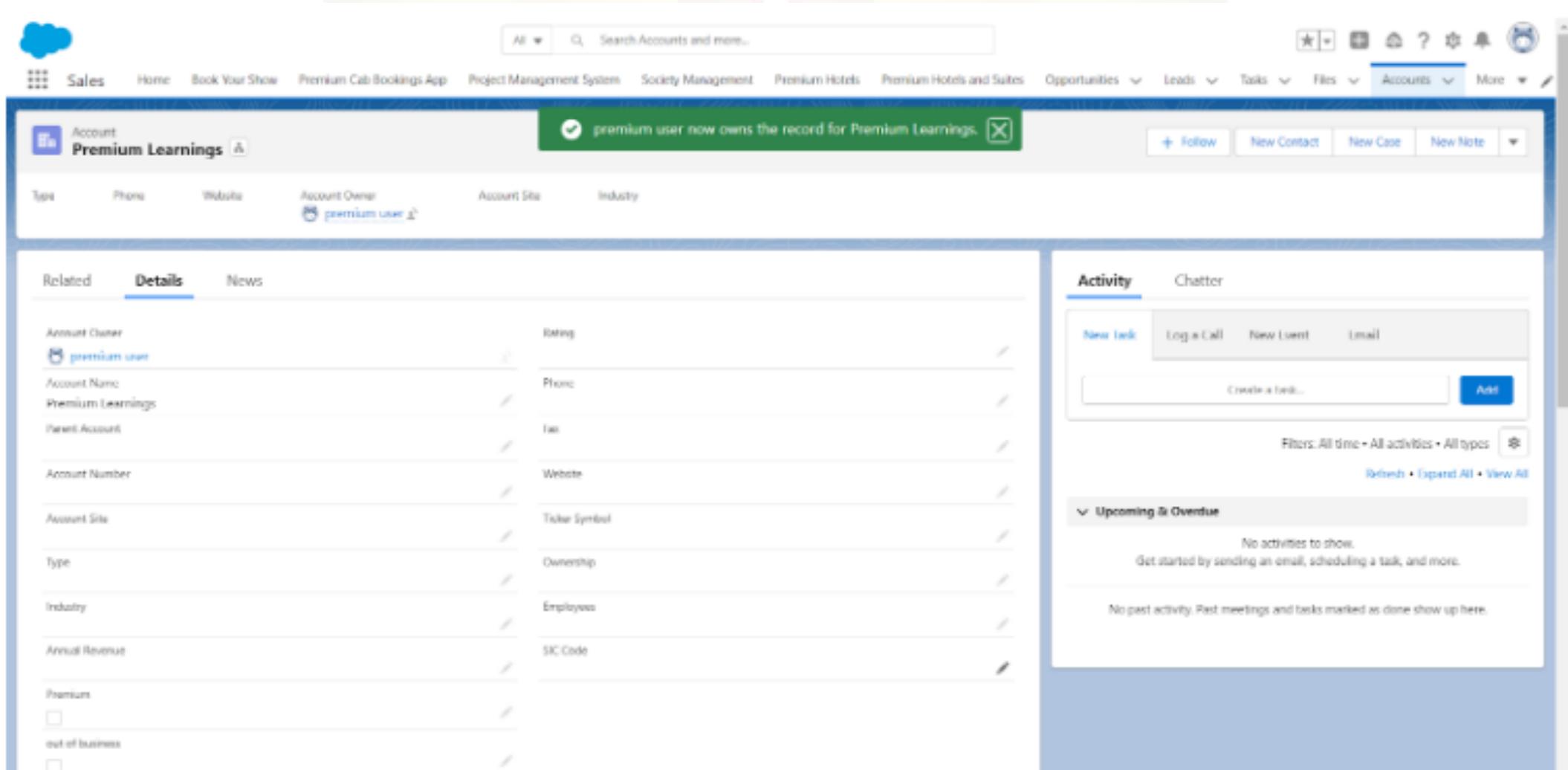
3. Now change the account owner of the same account record, and make any other user owner of the record (click on the account owner field and a pop up will appear to change owner)

select any other user apart from you and click change owner.

(If you cant see any other user in drop down list, create new user set up > home > quick find > users > new user > input the required fields and create a new user)



4. Now the new user is owner of record



4. Now check the Sales rep field and it will be updated with the newly assigned account owner as shown below

The screenshot shows the Microsoft Dynamics 365 Accounts page. At the top, there's a navigation bar with links like Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, and More. Below the navigation is a search bar with placeholder text "Search Accounts and more..." and a magnifying glass icon. The main content area displays account details for "Premium Learnings". The fields shown include:

- Type: Ownership
- Industry: Employees
- Annual Revenue: N/A
- Premium:
 - out of business
 - No of open opportunities
- Recent activity: None
- Total Opportunity Amount: N/A
- Client Contact: N/A
- Complete Opportunity Amount: N/A
- Sales Rep: premium user
- Count of Contacts: N/A
- No Of Contacts: N/A
- Billing Address: N/A
- Shipping Address: N/A

A sidebar on the right says "Get started by sending an email, scheduling a task, and more." and "No past activity. Past meetings and tasks marked as done show up here."

Example 21

Create the field called the “Contact Relationship” checkbox on the Contact Object and Create the related object called “Contact Relationship” which is a related list to the Contacts.(Look Up Relationship). Now logic is when we create contact by checking the Contact Relationship checkbox, then Contact Relationship will be created automatically for that contact.

CreateCRonContactCreation(apex trigger)

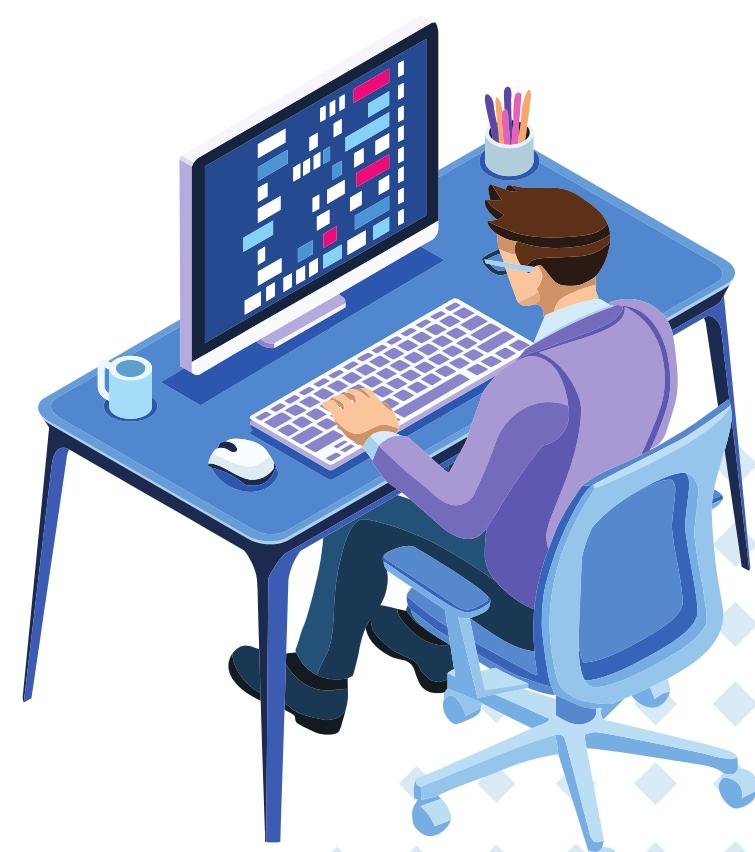
```
trigger CreateCRonContactCreation on Contact
(after insert){

    if(trigger.isAfter) {
        if(trigger.isInsert) {
            ContactMasterHandler ConIns = New
                ContactMasterHandler();
            ConIns.createContactRelationshipByContact(trigger.New);
        }
    }
}
```

ContactMasterHandler(apex class)

```
Public Class ContactMasterHandler_Undelete{
    public void undeleteContactRelationshipsByContact
        (list<Contact>List_Contacts)
    {
        set<Id> ContactIds = New set<Id>();
    }
}
```

```
if(List_Contacts!=null && List_Contacts.size()>0)
{
    list<Contact_Relationship__c> List_ConRels= new
        list<Contact_Relationship__c>();
    List_ConRels= [select id from Contact_
        Relationship__c where isDeleted=TRUE and
        Contact__c in:
        GlobalUtility.getUniqueIds(List_Contacts)];
    undelete List_ConRels;
}
}
```



Test Case:

1. Create a new contact as shown below

New Contact

Contact Information

Contact Owner  Sanjog Kawade	Phone <input type="text"/>
* Name <input type="text"/>	Home Phone <input type="text"/>
Salutation <input type="text"/>	
First Name <input type="text"/>	
* Last Name <input type="text" value="Premium Learnings"/>	
Account Name <input type="text"/>	Mobile <input type="text"/>
Title <input type="text"/>	Other Phone <input type="text"/>
Department <input type="text"/>	Fax <input type="text"/>
Birthdate <input type="text"/>	Email <input type="text"/>
Reports To <input type="text"/>	Assistant <input type="text"/>
<input type="button" value="Search Accounts..."/> <input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>	
Lead Source: --None--	
Asst. Phone: <input type="text"/>	

2. make sure the contact relationship checkbox is checked and save.

Search Accounts... <input type="text"/>	<input type="button" value=""/>
Title <input type="text"/>	Other Phone <input type="text"/>
Department <input type="text"/>	Fax <input type="text"/>
Birthdate <input type="text"/>	Email <input type="text"/>
Reports To <input type="text"/>	Assistant <input type="text"/>
Lead Source --None-- <input type="text"/>	Asst. Phone <input type="text"/>
Username <input type="text"/>	
Password <input type="text"/>	
Mobile <input type="text"/>	
Contact Relationship <input checked="" type="checkbox"/>	
Address Information Mailing Address Mailing Street <input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>	



The screenshot shows the Salesforce Contact page. At the top, there's a search bar with the placeholder "Search Contacts and more...". Below the search bar, a navigation bar includes links for Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Contacts, More, and a magnifying glass icon. A success message at the top right says "Contact [Mr. Premium Learnings] was created." The main content area displays a contact record for "Mr. Premium Learnings" with fields for Title, Account Name, Phone (2), Email, and Contact Owner (Sanjog Kawade). Action buttons at the bottom include "+ Follow", "New Case", "New Note", and "Submit for Approval".

3. Now check the contact relationship object and click to view “all” records

The screenshot shows the Salesforce Contact Relationships page. The top navigation bar is identical to the Contact page. The main content area features a "Recently Viewed" section with a list titled "JUST VISITED" containing "All" and "Recently Viewed (viewed list)". To the right is a search bar and a toolbar with buttons for New, Import, Change Owner, and Print View. Below this is a larger search bar labeled "Search this list..." and a set of filter icons.

4. You will see a record with same name as contact we created above, automatically created

The screenshot shows the Salesforce Contact Relationships page. The top navigation bar is identical to the Contact page. The main content area displays a list of contact relationships. The first item is "Customer Test User Customer Test user" and the second item is "Premium Learnings". There are checkboxes next to each item. A status message at the top left says "2 items - Sorted by Contact Relationship Name - Filtered by All contact relationships - Updated a few moments ago". To the right is a search bar and a toolbar with buttons for New, Import, Change Owner, and Print View. Below this is a larger search bar labeled "Search the list..." and a set of filter icons. At the bottom of the page, a URL is visible: "https://premiumlearnings-9b-dev-ed.lightning.force.com/lightning/l/a/c1/00000/pg/0AAA/.."

Example 22

When we change the Owner of the Contact Relationship, then the Owner name will be automatically populated in the Contact Relationship Name field.

ContactRelationshipMasterTrigger(apex trigger)

```
trigger ContactRelationshipMasterTrigger on
Contact_Relationship__c(before update){

if(trigger.isBefore){
    if(trigger.isUpdate){
        //call the handler for the before update trigger event
        updateCROwnerName ConRelUpd = New updateCROwnerName();
        ConRelUpd.updateContactRelationshipNameByOwner(trigger.New);
    }
}
}
```

updateCROwnerName (apex class)

```
Public Class updateCROwnerName{
    public void updateContactRelationshipNameByOwner
        (list<Contact_Relationship__c> cont_Rel) {
    map<Id, Id> map_Id_Own = new map<id,id>();
    map<Id, string> map_id_Name = new map<id,string>();
    set<id> Idset = new set<id>();
    for(Contact_Relationship__c List_recs:cont_Rel)
    {
        Idset.add(List_recs.Ownerid);
    }
    list<user> u=[select id,Name from user where id in:Idset];
    for(user list_users:u)
    {
        map_id_Name.put(list_users.Id,list_users.Name);
    }
}
```

```
if(u!=null && u.size()>0)
{
    for(Contact_Relationship__c List_recs:cont_Rel)
    {
        if (List_recs.Ownerid!=null)
        {
            List_recs.Name =
                map_id_Name.get(List_recs.Ownerid);
        }
    }
}
```



Test Case:

- After you create a contact relationship object, create a new record for it. And save

New Contact Relationship

Information

*Contact Relationship Name	Owner
Premium Learnings	Sanjog Kawade
Contact	
Search Contacts...	<input type="button" value=""/>

Cancel Save & New Save

- The record is saved.(notice owner name and contact relationship name)

Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Contact Relationships More

Contact Relationship Premium Learnings Contact Relationship Premium Learnings was created.

Related	Details
Contact Relationship Name Premium Learnings	Owner Sanjog Kawade
Contact	
Created By Sanjog Kawade, 7/8/2022, 12:20 PM	Last Modified By Sanjog Kawade, 7/8/2022, 12:20 PM

Activity

New Event New Task Log a Call Email

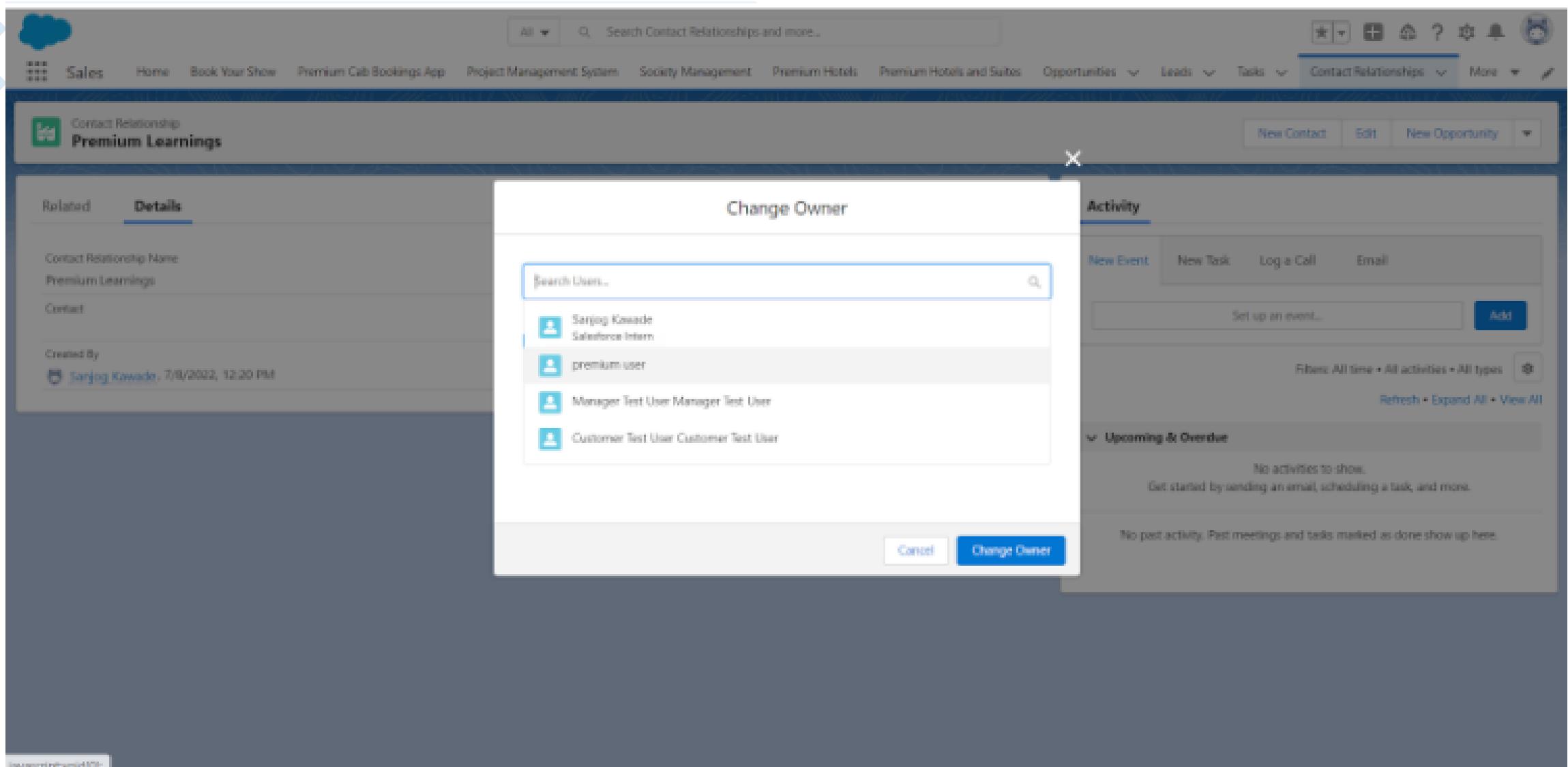
Set up an event... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

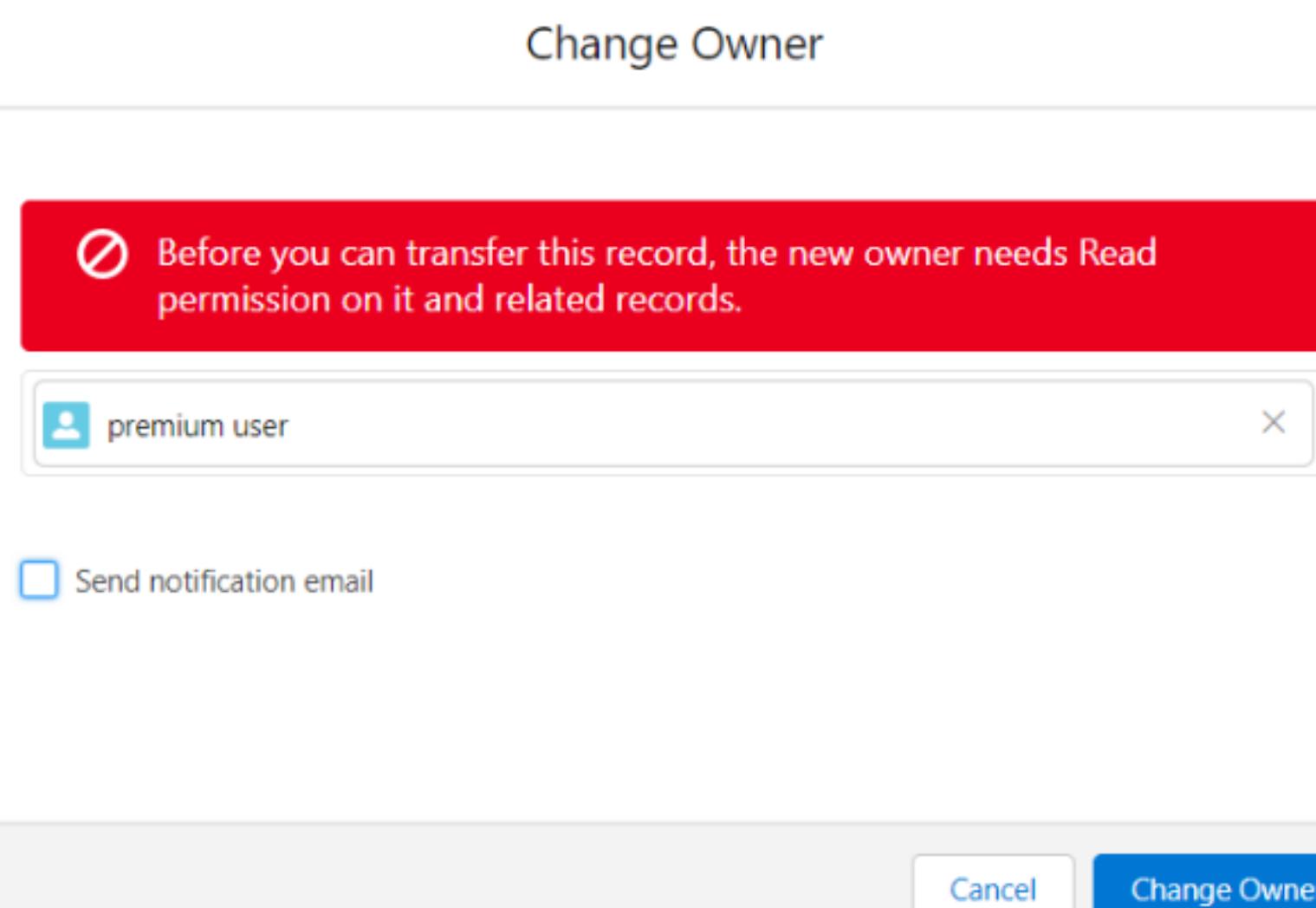
No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

3. Now edit the record to change the owner name and make any other user owner of the record.



4. If you get the following error, go to setup > home > quick find > users > select user name that you wish to make owner of this record(customer test user in our case) > check the profile name that user is associated with(in our case it is customer)



The screenshot shows the Salesforce Setup interface under the 'Users' section. The left sidebar includes options like 'Permission Set Groups', 'Profiles', 'Public Groups', 'Queues', 'Roles', 'User Management Settings', 'Users' (which is selected), 'Feature Settings', 'Data.com', 'Prospector', 'User Interface', 'Action Link Templates', 'Actions & Recommendations', 'App Menu', 'Custom Labels', 'Density Settings', 'Global Actions', 'Global Actions', and 'Publisher Layouts'. The main content area displays a table titled 'All Users' with columns for 'Action', 'Full Name', 'Alias', 'Username', 'Role', and 'Active'. The table lists several users, including 'Chatter Expert', 'Customer Test User', 'Customer Test User', 'Kwade_Saqing', 'Manager Test User', 'Manager Test User', 'User_Inspration', 'User_premium', and 'User_Security'. The 'Active' column shows checkboxes for each user, and the 'Profile' column lists their respective profiles: 'Chatter Free User', 'Customer', 'System Administrator', 'Manager', 'Analytics Cloud Integration User', 'Force.com - App Subscription User', and 'Analytics Cloud Security User'. Navigation links at the bottom include 'New User', 'Reset Password(s)', and 'Add Multiple Users'.

5. Now go to setup > home > quick find > profile > search the profile customer. Scroll down to find custom object permissions

The screenshot shows the Salesforce Setup interface under the 'Profiles' section. The left sidebar is identical to the previous screenshot. The main content area shows the 'Custom Object Permissions' section. It features a grid of objects on the left and permission levels on the right. The objects listed include Airlines, Award, Banks, Bookings, BookingI, Cab Customers, Cab Reservations, Cities, Clients, Consultants, Contact Relationships, customers, Customers, Customer2, Customer Projects, Delivery Confirmation, Delivery Customers, Flights, Line Items, Maintenance, Members, Offers, Passengers, Past Information, Projects, References, Requirements, Reservations, Risers, scoreCards, Staffs, Stakeholders, Tasks Assigned, Test Objects, and Tickets. Each object has a row of checkboxes for 'Basic Access' (Read, Create, Edit, Delete) and 'Data Administration' (View All, Modify All).



6. Find contact relationship object and give read create edit and delete permissions(tick the respective boxes) and click save.

	Read	Create	Edit	Delete	View All	Modify All
Actions	<input type="checkbox"/>					
Analyst	<input type="checkbox"/>					
Booker	<input type="checkbox"/>					
Booking	<input type="checkbox"/>					
Cab Customer	<input type="checkbox"/>					
Cab Reservation	<input checked="" type="checkbox"/>					
Clerk	<input type="checkbox"/>					
Clients	<input type="checkbox"/>					
Consorts	<input type="checkbox"/>					
Customer Relationship	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Customers	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Customer Projects	<input type="checkbox"/>					
Delivery Confirmation	<input type="checkbox"/>					
Delivery Customers	<input type="checkbox"/>					
Delivery Drivers	<input type="checkbox"/>					
Delivery Items	<input type="checkbox"/>					
Drivers	<input type="checkbox"/>					
Employee	<input type="checkbox"/>					
User Roles	<input type="checkbox"/>					
Manufacture	<input type="checkbox"/>					
Members	<input type="checkbox"/>					
Offices	<input type="checkbox"/>					
Passenger	<input type="checkbox"/>					
Per Information	<input type="checkbox"/>					
Project	<input type="checkbox"/>					
Release	<input type="checkbox"/>					
Requirements	<input type="checkbox"/>					
Reservations	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Rewards	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Scorecards	<input type="checkbox"/>					
Staffs	<input type="checkbox"/>					
Salesholders	<input type="checkbox"/>					
Tasks Assigned	<input type="checkbox"/>					
Test Objects	<input type="checkbox"/>					
Top X Designations	<input type="checkbox"/>					
Trainings	<input type="checkbox"/>					
Vehicles	<input type="checkbox"/>					

7. Now select the user as owner as explained in step 3 and change owner

Change Owner

Customer Test User Customer Test User

Send notification email

Cancel Change Owner

Activity

New Event New Task Log a Call Email

Set up an event... Add

Filter: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

8. Now notice the owner name and customer relationship name

The screenshot shows a CRM application interface. At the top, there's a navigation bar with icons for Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Contact Relationships, and More. A search bar is also present.

The main area displays a contact relationship record titled "Customer Test User Customer Test User". The "Details" tab is selected. Key fields shown include:

- Contact Relationship Name: Customer Test User Customer Test User
- Owner: Customer Test User Customer Test User
- Created By: Sanjog Kavade, 7/8/2022, 12:20 PM
- Last Modified By: Sanjog Kavade, 7/8/2022, 12:46 PM

To the right, there's an "Activity" section with tabs for New Event, New Task, Log a Call, and Email. It includes a button to "Set up an event..." and a link to "Add". Below this, there are sections for "Upcoming & Ongoing" and "Past activity".



Example 23

Create field called “Count of Contacts” on Account Object. When we add the Contacts for that Account then the count will populate in the field on the Account details page. When we delete the Contacts for that Account, the Count will update automatically.

```
trigger CountOfContacts on Contact (after insert,
after delete){

    set<id> accid=new set<id>();
    list<contact> contactlist =new list<contact>();
    list<contact> listcon=new list<contact>();
    list<account> acclist=new list<account>();
    list<account> listAcc=new list<account>();
    map<id,integer> mapCount=new map<id,integer>();

    if(trigger.isinsert){
        for(contact con:trigger.new){
            accid.add(con.accountid);
        }
    }

    if(trigger.isdelete){
        for(contact con:trigger.old)
        {
            accid.add(con.accountid);
        }
    }

    acclist=[select id,name from account where id in:accid];
    contactlist = [select id,name,accountid from contact
                  where accountid in:accid];
}
```

```
for(account acc:acclist)
{
    listcon.clear();
    for(contact c:contactlist){
        if(c.accountid==acc.id){
            listcon.add(c);
            mapCount.put(c.accountid,listcon.size());
        }
    }
}

if(acclist.size()>0){
    for(Account a:acclist){
        if(mapCount.get(a.id)==null)
            a.Count_of_Contacts__c=0;
        else
        {
            a.Count_of_Contacts__c=mapCount.get(a.id);
            listAcc.add(a);
        }
    }
}

if(listAcc.size()>0)
    update listAcc;
}
```



Test Case:

1. Create an account record without any contacts related to the record and click save. Notice the no of contacts field is blank.

Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Accounts More

Premium Learnings

Reason:
out of business
No of open opportunities

Revenue Type:
Total Opportunity Amount
Client Contact
Complete Opportunity Amount

Sales Rep:
premium user
Count of Contracts

No Of Contacts

Billing Address Shipping Address

Customer Priority SLA

SLA Expiration Date SLA Serial Number

Number of Locations

2. Now go to the related lists of the account and notice there are no contacts related to this account

Sales Home Book Your Show Premium Cab Bookings App Project Management System Society Management Premium Hotels Premium Hotels and Suites Opportunities Leads Tasks Files Accounts More

Premium Learnings

Type Phone Website Account Owner **premium user** Account Site Industry

Related Details News

We found no potential duplicates of this Account.

Contacts (0) **New**

Opportunities (0) **New**

Cases (0) **New**

Notes & Attachments (0) **Upload files**
Or drag files

Partners (0) **New**

Activity Chatter

New Task Log a Call New Event Email

Create a task... Add

Filter: All time • All activities • All types Refresh Expand All View All

Upcoming & Overage

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

3. Add a contact in the related lists by clicking new button; fill in all the details and click save.

The screenshot shows the Salesforce Account page for 'Premium Learnings'. The top navigation bar includes Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, and More. A green success message at the top right says 'Contact "Mr. Premium Learnings" was created.' The main content area has tabs for Related, Details, and News. The Related tab shows a list of Contacts (1), Opportunities (0), Cases (0), and Notes & Attachments (0). The Activity tab shows sections for Activity and Chatter, with options to Create Task, Log a Call, New Event, and Email. It also displays 'Upcoming & Overdue' tasks.

4. Now check the account details, and the field count of contacts. It will show the value 1

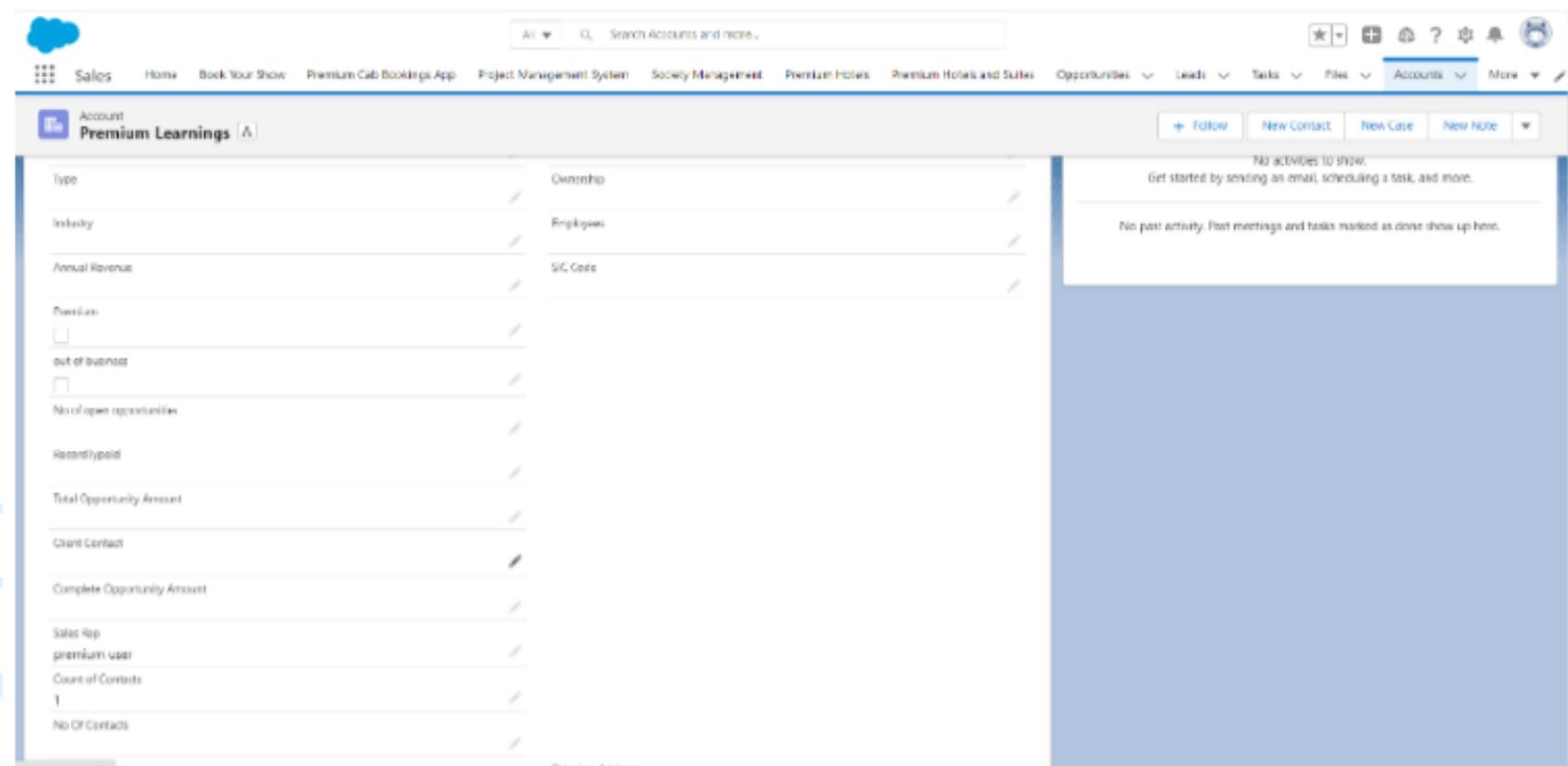
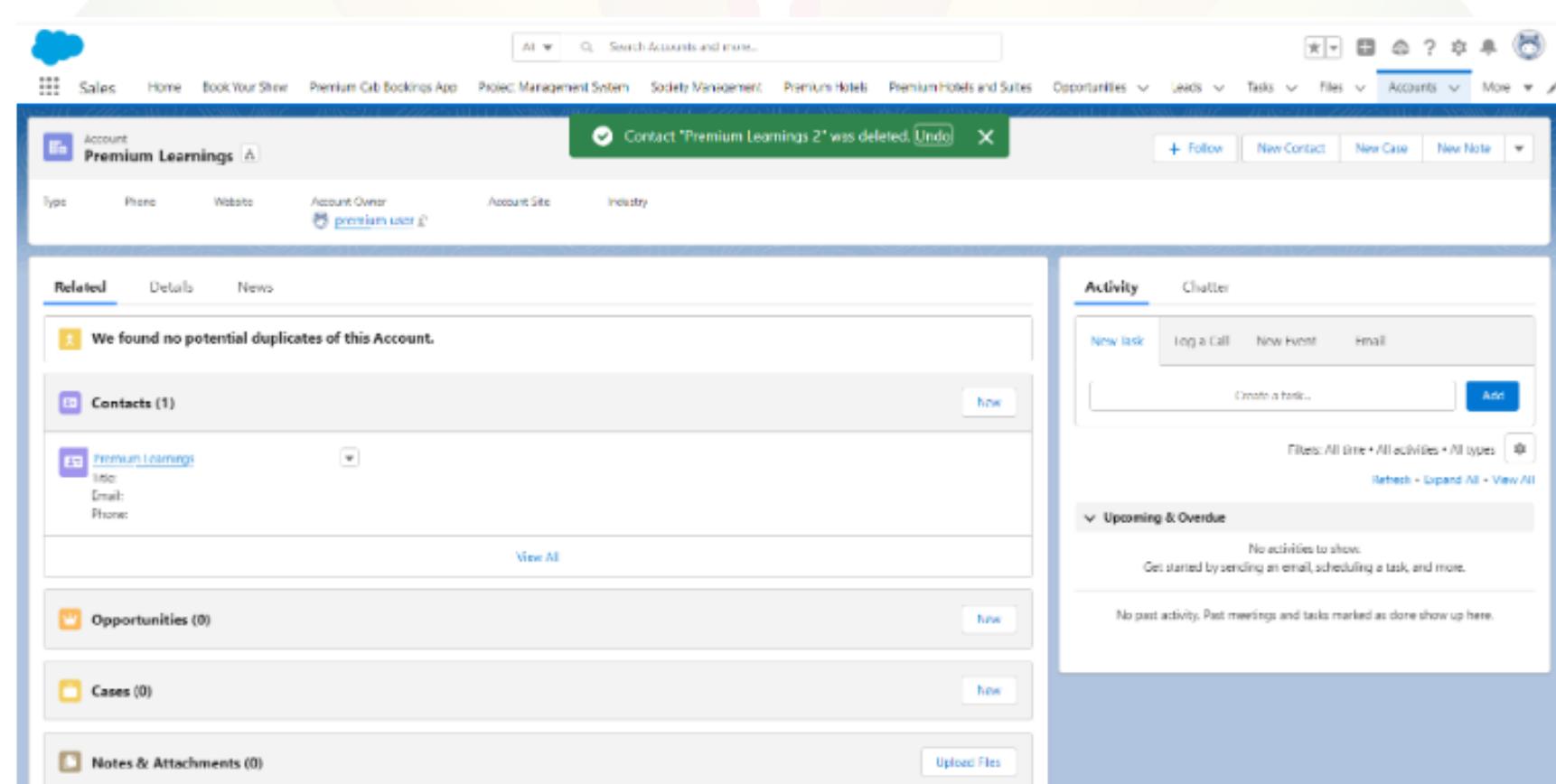
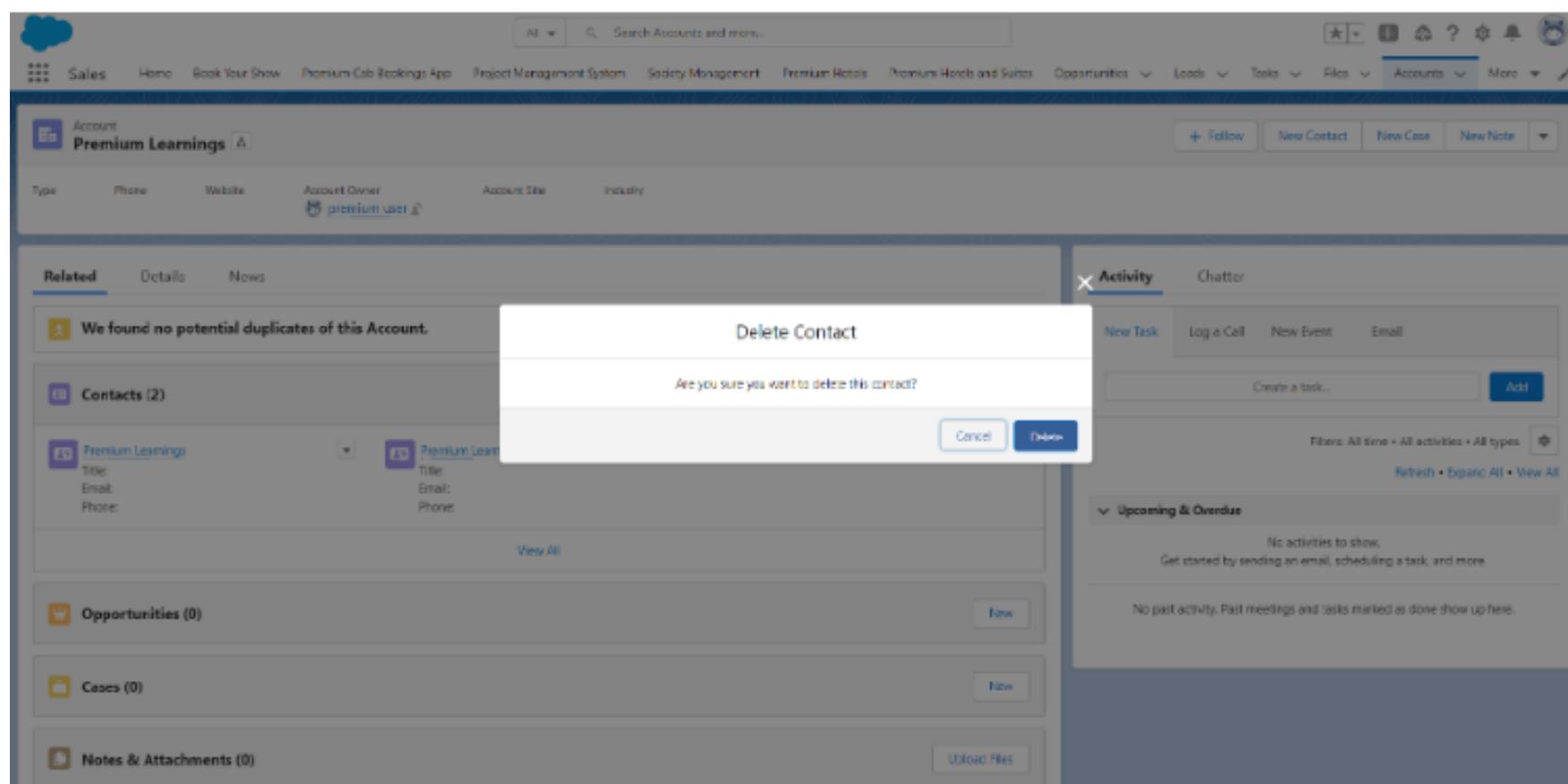
The screenshot shows the detailed view of the 'Premium Learnings' account. The left sidebar lists fields: Industry, Annual Revenue, Premium, out of business, No of open opportunities, RecordType, Total Opportunity Amount, Client Contact, Complete Opportunity Amount, Sales Rep, Count of Contacts (1), No Of Contacts, Billing Address, and SLA. The right panel shows the Activity section with a message: 'No past activity. Past meetings and tasks marked as done show up here.'

5. repeat step 3 and add another contact, check the count of contactsfield now getting updated to 2

The screenshot shows the Salesforce Account page for 'Premium Learnings'. At the top, a green success message says 'Contact "Premium Learnings 2" was created.' Below the header, there are tabs for 'Related', 'Details', and 'News'. Under 'Related', there is a section for 'Contacts (2)' which lists 'Premium Learning' and 'Premium Learnings 2'. The 'Premium Learnings 2' contact has fields for Title, Email, and Phone. To the right of the account details is an 'Activity' sidebar with sections for 'New Task', 'Log a Call', 'New Event', and 'Email'. It also includes a 'Create a task...' input field and a 'Chatter' section.

This screenshot shows the same Account page for 'Premium Learnings' after adding another contact. The 'Count of Contacts' field in the 'Details' section is now updated to '2'. The rest of the page content is identical to the previous screenshot.

6. Now delete one of the just created contacts and check the count of contacts field getting updated to 1 again





Example 24

Create the object called “Customer12” and create the Master-Detail Relationship on Customer12 object so that Customer12 will be the related list to the Account record. Create the field called “Account Manager” on the Customer12 object which is a lookup to the user object. Now Logic is when we create a Customer12 record for an account record, then the user in the Account Manager field will be automatically added to the Account Team of that associated account.

pre-requisite:

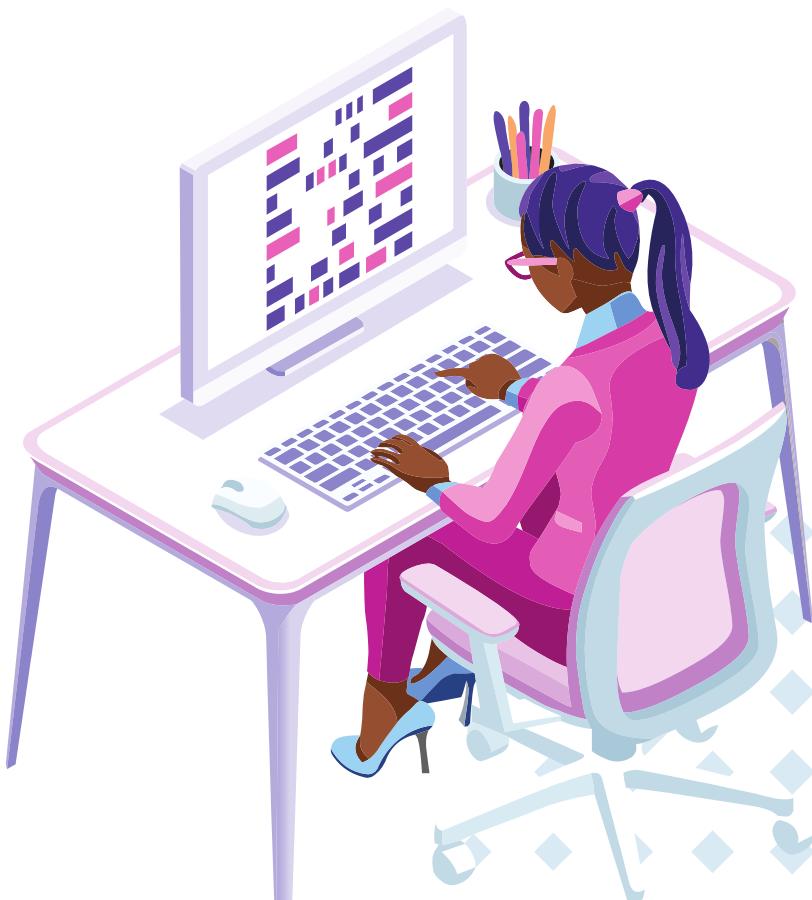
From Setup, enter Account Teams in the Quick Find box, and select Account Teams. Click Enable Account Teams. Select the Account Teams Enabled checkbox and click Save. Select the Account Layout checkbox to add the Account Team related list to the page layout. Select the Add to users' customized related lists checkbox and click Save.

```
trigger InsertAccountTeam on Customer12__c (after insert) {

    List<AccountTeamMember> atm_list=new
                                List<AccountTeamMember>();
    AccountTeamMember atm = new AccountTeamMember();
    List<AccountShare> newShare = new List<AccountShare>();

    if(trigger.isInsert) {
        For(Customer12__c c:Trigger.new)
        {
            if(c.AccountManager__c!=null)
            {
                atm = new AccountTeamMember();
                atm.accountid=c.Account__c;
                atm.teamMemberRole='Account Manager';
                atm.UserId=c.AccountManager__c;
                AccountShare shares = new AccountShare();
                shares.AccountId=c.Account__c;
```

```
shares.UserOrGroupId = c.AccountManager__c;
shares.AccountAccessLevel='Read/Write';
shares.OpportunityAccessLevel = 'Read Only';
shares.CaseAccessLevel='Read Only';
newShare.add(shares);
atm_list.add(atm);
}
}
if(atm_list!=null)
insert atm_list;
if(newShare!=null && newShare.size()>0)
List<Database.saveresult>sr=
Database.insert(newShare,false);
}
}
```





Test Case:

1. Create an account named 'Premium Learnings' and save it.

The screenshot shows the Salesforce Account creation page. The account name is 'Premium Learnings'. The left sidebar contains fields for Revenue, Client Contact, and SLA information. The right side of the page is a large blue placeholder area.

2. Now check the related lists to see the account teams, which will be blank.

The screenshot shows the Salesforce Account page for 'Premium Learnings'. The related lists section is displayed, showing the following counts: Notes & Attachments (0), Partners (0), consultants (0), customers (0), Assets (0), Account Team (0), and Customers12 (0). The 'Account Team' section is highlighted, showing 0 members.



3. Now create a Customer12 record with name 'Premium'. In the account field, select the premium learnings account we created in above step. And in account manager field select your name and click save.

New Customer12

Information

* Customer12 Name	Premium
* Account	Premium Learnings
Account Manager	Sanjog Kawade

Cancel Save & New Save

4. Now see the related lists of premium learnings account again, and see the account team; you will find your name(or the account manager you selected in step 3) being populated in the account team.

The screenshot shows the Salesforce interface for the 'Premium Learnings' account. On the left, there's a sidebar with various tabs like Sales, Home, Book Your Show, etc. The main content area shows the account details. Under 'Related Lists', there's a section for 'Account Team (1)' which lists 'Sanjog Kawade' as the Account Manager. Below that, there's a section for 'Customers12 (1)' which lists a record named 'Premium'. The top navigation bar includes buttons for Follow, New Contact, New Case, and New Note.



Example 25

The above trigger Logic(Example 24) is when we create a Customer12 record for an account record, then the user in the Account Manager field will be automatically added to the Account Team of that associated account. Now the following trigger logic is when we update the user in the “Account Manager”, the Account team will be updated automatically.

pre-requisite:

From Setup, enter Account Teams in the Quick Find box, and select Account Teams. Click Enable Account Teams. Select the Account Teams Enabled checkbox and click Save. Select the Account Layout checkbox to add the Account Team related list to the page layout. Select the Add to users' customized related lists checkbox and click Save.

```
trigger UpdateAccountTeam on Customer12_c (before update) {

    List<AccountTeamMember> atm_list=new
                                List<AccountTeamMember>();
    AccountTeamMember atm = new AccountTeamMember();
    List<AccountShare> newShare = new List<AccountShare>();

    if(trigger.isupdate)
    {
        if(trigger.isbefore)
        {
            Set<Id> setAccIds1=new Set<Id>();
            Set<Id> setDelATM=new Set<Id>();
            Map<id,Set<Id>> mapAccMgrs=new Map<id,Set<Id>>();
            for(Customer12_c c:Trigger.new)
            {
                if(trigger.oldmap.get(c.Id).AccountManager__c !=
                    c.AccountManager__c && c.AccountManager__c!=null)
                {
                    setAccIds1.add(c.Account__c);
                }
            }
        }
    }
}
```

```
List<Customer12__c> listPLAccMgrs=[select id,
    AccountManager__c,Account__c from Customer12__c
    where Account__c in :setAccIds1 and id not
    in :trigger.newmap.keySet()];
```

```
if(listPLAccMgrs!=null && listPLAccMgrs.size()>0)
{
    for(Customer12__c c:listPLAccMgrs)
    {
        Set<Id> idMgrs=mapAccMgrs.get(c.Account__c);
        if(null==idMgrs){
            idMgrs=new set<Id>();
            mapAccMgrs.put(c.Account__c,idMgrs);
        }
        idMgrs.add(c.AccountManager__c);
    }
}
```

```
Map<id,List<AccountTeamMember>> mapStdAccTeam=new
Map<id,List<AccountTeamMember>>();
List<AccountTeamMember> listStdAccTeam=[select id,
    UserId,AccountId from AccountTeamMember
    where AccountId in :setAccIds1];
```

```
if(listStdAccTeam!=null && listStdAccTeam.size()>0){
    for(AccountTeamMember recAccTeam :listStdAccTeam)
    {
        List<AccountTeamMember> listStdAccTeamMap=
            mapStdAccTeam.get(recAccTeam.AccountId);
        if(null==listStdAccTeamMap)
        {
            listStdAccTeamMap=new List<AccountTeamMember>();
            mapStdAccTeam.put(recAccTeam.AccountId,listStdAccTeamMap);
        }
    }
}
```

```

listStdAccTeamMap.add(recAccTeam);
}

}

system.debug('*****'+mapAccMgrs);

for(Customer12__c c:Trigger.new)
{

if(trigger.oldmap.get(c.Id).AccountManager__c !=
    c.AccountManager__c && c.AccountManager__c!=null )
{

List<AccountTeamMember>listAccTeam=
    mapStdAccTeam.get(c.Account__c);
Set<Id> idMgrs=mapAccMgrs.get(c.Account__c);

if(listAccTeam!=null && listAccTeam.size()>0 )
{
    if(idMgrs!=null && idMgrs.size()>0 && !
        (idMgrs.Contains(trigger.oldmap.get(c.Id)
                           .AccountManager__c)))
    {
        for(AccountTeamMember recATM:listAccTeam)
        {
            if(recATM.UserId==trigger.oldmap.get(c.Id)
                           .AccountManager__c)
                setDelATM.add(recATM.Id);
        }
    }
    else if(idMgrs==null)
    {
        for(AccountTeamMember recATM:listAccTeam)
            setDelATM.add(recATM.Id);
    }
}
}

```

```

atm = new AccountTeamMember
        (accountid=c.Account__c,teamMemberRole='Account
         Manager',UserId=c.AccountManager__c);
AccountShare shares = new AccountShare();
shares.AccountId=c.Account__c;
shares.UserOrGroupId = c.AccountManager__c;
shares.AccountAccessLevel='Edit';
shares.OpportunityAccessLevel = 'None';
newShare.add(shares);
atm_list.add(atm);
}
}

```

```

List<AccountTeamMember> listDelATM=[select id from
AccountTeamMember where id in:setDelATM];

if(listDelATM!=null && listDelATM.size()>0 )
{
    delete listDelATM;
    if(atm_list!=null)
    {
        insert atm_list;
        if(newShare!=null && newShare.size()>0)
            List<Database.saveresult> sr=
                Database.insert(newShare,false);
    }
}
}

```



Test Case:

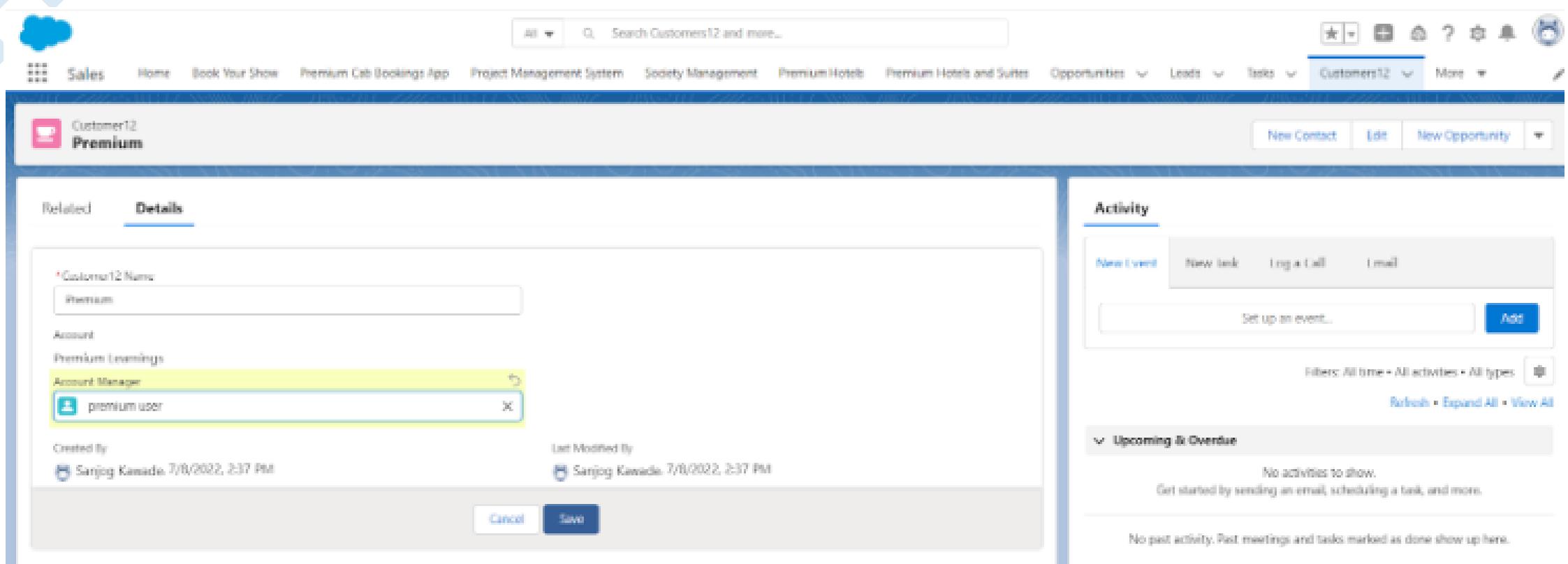
1. Create an account record named 'Premium Learnings'

The screenshot shows the Salesforce Account page for 'Premium Learnings'. The page has a header with navigation links like Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, and More. Below the header, there's a search bar and a toolbar with buttons for Follow, New Contact, New Case, and New Note. The main content area has tabs for Related, Details, and News. Under the Details tab, there are fields for Account Owner (Sanjog Kawade), Type (Phone), Website, Account Name (Premium Learnings), Account Site, Industry, Rating, Phone, Fax, Website, Ticker Symbol, Ownership, Employees, Annual Revenue, SIC Code, Premium (checkboxes for Out of Business and Premium), and a Notes section. To the right, there's an Activity sidebar with tabs for Activity and Chatter. The Activity tab shows buttons for New Task, Log a Call, New Event, and Email, and a text input field for 'Create a task...'. It also includes filters for All time, All activities, All types, Refresh, Expand All, and View All.

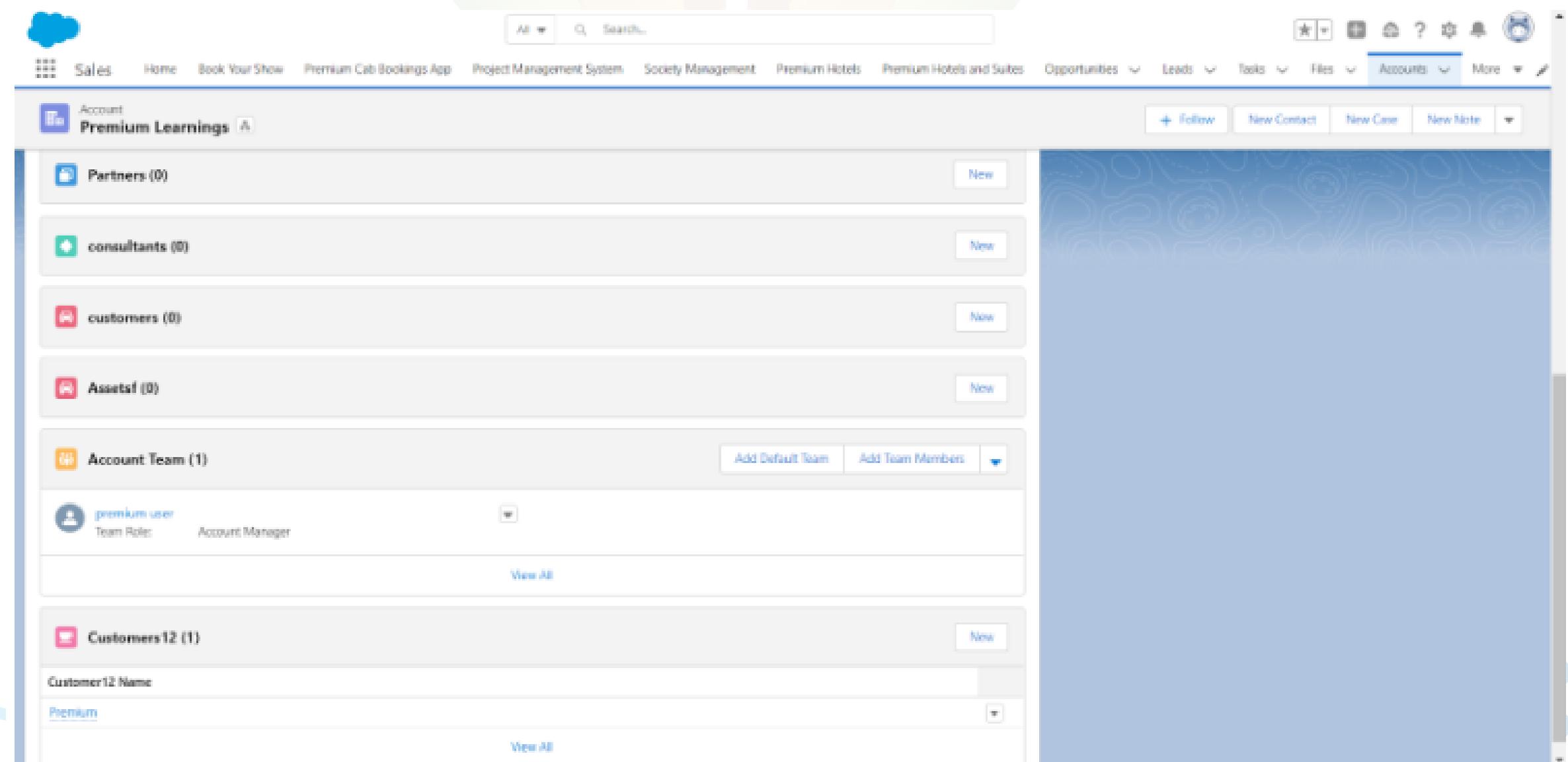
2. create a Customer12 record as shown below. (Put your name/your orgs system administrator in account manager field). Select the account we created above in the account field of the record.

The screenshot shows the Salesforce Customer12 page for 'Premium'. The page has a header with buttons for New Contact, Edit, and New Opportunity. Below the header, there's a toolbar with buttons for New Item, New Task, Log a call, and Email. The main content area has tabs for Related and Details. Under the Details tab, there are fields for Customer12 Name (Premium), Account (Premium Learnings), Account Manager (Sanjog Kawade), and Created By (Sanjog Kawade, 7/8/2022, 2:37 PM). There's also a Last Modified By field showing Sanjog Kawade, 7/8/2022, 2:37 PM. To the right, there's an Activity sidebar with tabs for Activity and Chatter. The Activity tab shows buttons for New Item, New Task, Log a call, and Email, and a text input field for 'Get up an event...'. It also includes filters for All time, All activities, All types, Refresh, Expand All, and View All.

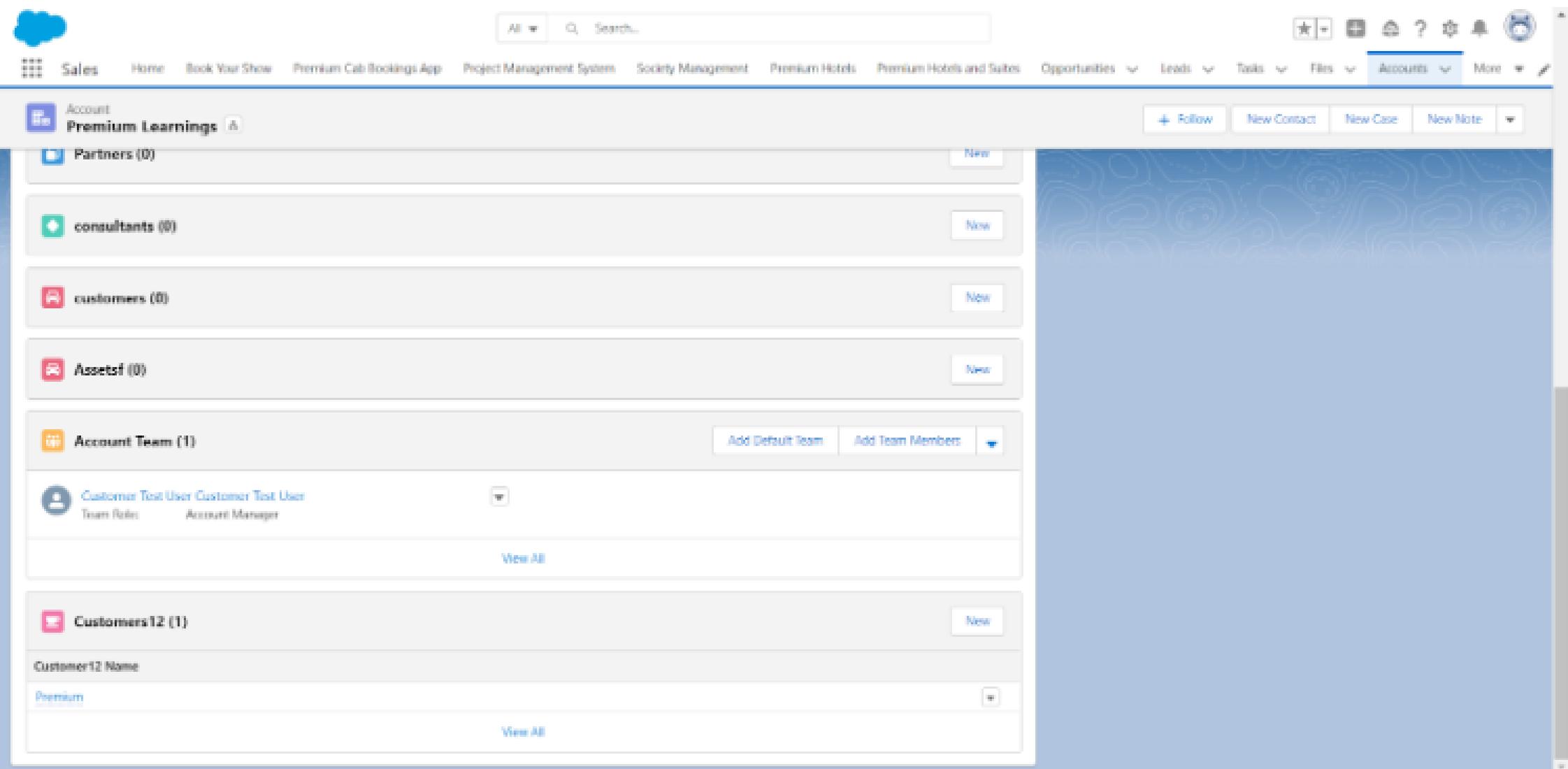
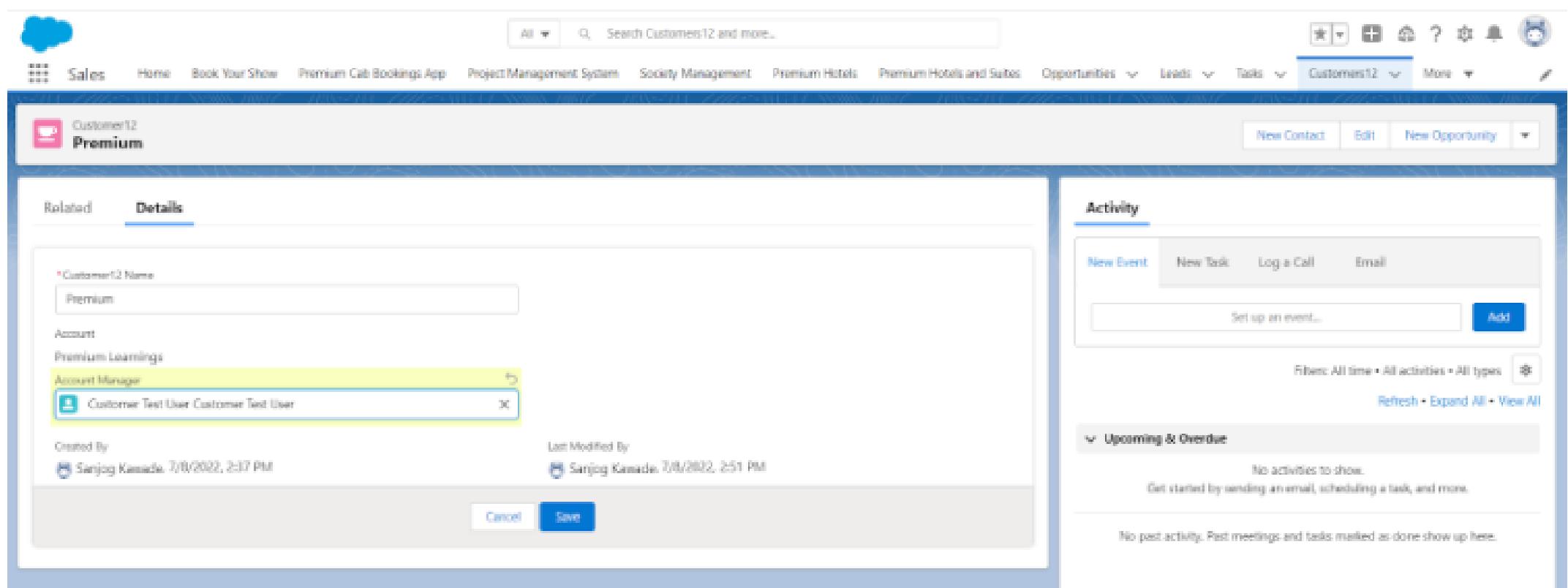
3. Now Update the account manager field and select any other user instead of the current one. Click Save.



4. Now check related list of account we created in step one, and find account team in it. We will see the account manager(premium user) we updated above in it.



5. Repeat steps 3 and 4 again with a different user as account manager and see the account team field is updated as well.



Example 26

The trigger scenario 35 Logic is when we create a Customer12 record for the account record, then the user in the Account Manager field will be automatically added to the Account Team of that associated account. Now the following trigger gives the logic about when we delete the “Customer12” of that account, then the user will be deleted automatically from the Account Team of that account.

pre-requisite:

From Setup, enter Account Teams in the Quick Find box, and select Account Teams. Click Enable Account Teams. Select the Account Teams Enabled checkbox and click Save. Select the Account Layout checkbox to add the Account Team related list to the page layout. Select the Add to users' customized related lists checkbox and click Save.

```
trigger DeleteAccountTeam on Customer12__c (before delete)
{
    List<AccountTeamMember> atm_list=new
    List<AccountTeamMember>();
    AccountTeamMember atm = new AccountTeamMember();
    List<AccountShare> newShare = new List<AccountShare>();

    if(trigger.isdelete)
    {
        set<id> setAccids = new set<id>();
        Set<Id> setDelATM=new Set<Id>();
        Map<id,Set<Id>> mapAccMgrs=new Map<id,Set<Id>>();
        for(Customer12__c c:Trigger.old)
        {
            setAccids.add(c.Account__c);
        }
        List<Customer12__c> listPLAccMgrs=[select id,
            AccountManager__c,Account__c from Customer12__c
            where Account__c in:setAccids and id not
            in:trigger.oldmap.keySet()];
    }
}
```



```

if(listPLAccMgrs!=null && listPLAccMgrs.size(>0)
{
    for(Customer12__c c:listPLAccMgrs)
    {
        Set<Id> idMgrs=mapAccMgrs.get(c.Account__c);
        if(null==idMgrs){
            idMgrs=new set<Id>();
            mapAccMgrs.put(c.Account__c,idMgrs);
        }
        idMgrs.add(c.AccountManager__c);
    }
}

```

```

Map<id,List<AccountTeamMember>> mapStdAccTeam=new
    Map<id,List<AccountTeamMember>>();
List<AccountTeamMember> listStdAccTeam=[select id,
    UserId,AccountId from AccountTeamMember
    where AccountId in:setAccids];

if(listStdAccTeam!=null && listStdAccTeam.size(>0)
{
    for(AccountTeamMember recAccTeam :listStdAccTeam)
    {
        List<AccountTeamMember>
        listStdAccTeamMap =
            mapStdAccTeam.get(recAccTeam.AccountId);
    }
}

```

```

        if(null==listStdAccTeamMap)
        {
            listStdAccTeamMap=new List<AccountTeamMember>();
            mapStdAccTeam.put(recAccTeam.AccountId.
                listStdAccTeamMap);
        }
        listStdAccTeamMap.add(recAccTeam);
    }
}

for(Customer12__c c:Trigger.old)
{
    List<AccountTeamMember>
    listAccTeam=mapStdAccTeam.get(c.Account__c);
    Set<Id> idMgrs=mapAccMgrs.get(c.Account__c);

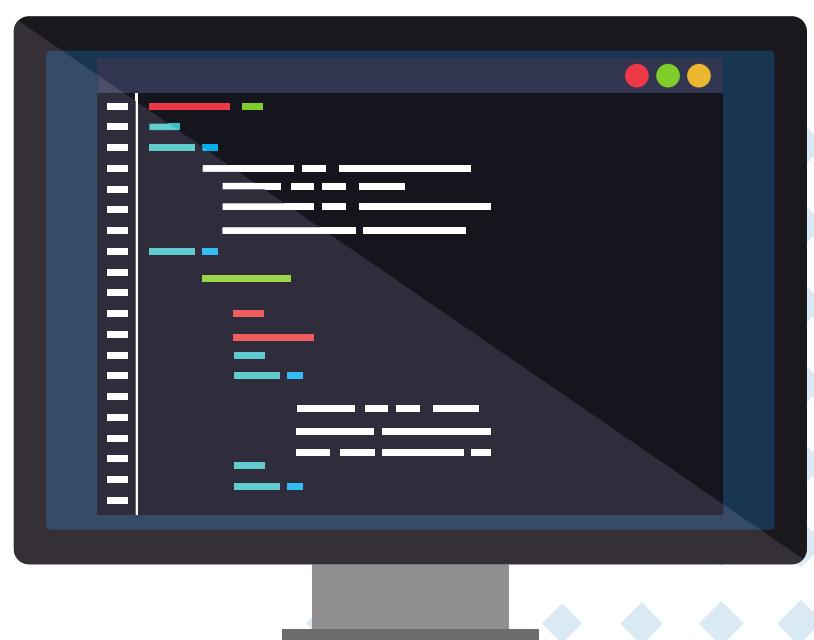
    if(listAccTeam!=null && listAccTeam.size()>0 )
    {
        if(idMgrs!=null && idMgrs.size()>0 &&
           !(idMgrs.Contains(trigger.oldmap.get(c.Id)
               .AccountManager__c)))
        {
            for(AccountTeamMember recATM:listAccTeam)
            {
                if(recATM.UserId ==
                    trigger.oldmap.get(c.Id).AccountManager__c)
                    setDelATM.add(recATM.Id);
            }
        }
    }
}

```

```
        else if(idMgrs==null)
        {
            for(AccountTeamMember recATM:listAccTeam)
                setDelATM.add(recATM.Id);
        }
    }

List<AccountTeamMember> listDelATM=[select id from
                                         AccountTeamMember where id in:setDelATM];

if(listDelATM!=null && listDelATM.size(>0 )
delete listDelATM;
}
```



Test Case:

1. In the previous examples we created a customer12 record named 'premium' with account manager field populated with a user. The same user will be populated in the account team (in related list) of the lookup account of customer12.

Customer12
Premium

Related **Details**

Customer12 Name: Premium
Account Manager: Customer Test User

Activity

New Event | New Task | Log a Call | Email

Set up an event... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

Account
Premium Learnings

Related

Partners (0) | New

customers (0) | New

Assets (0) | New

Account Team (1)

Add Default Team | Add Team Members

Customer Test User Customer Test User
Team Role: Account Manager

View All

Customers12 (1)

New

Customer12 Name: Premium

View All



2. Now delete the customer12 record named 'Premium' we created

The screenshot shows the Salesforce Sales tab interface. A modal dialog titled "Delete Customer12" is open, asking "Are you sure you want to delete this Customer12?". Below the dialog are two buttons: "Cancel" and "Delete". The background shows the account details for "Customer12 Premium", including its name, account number, and creation date by "Sergey Kovalski". To the right of the account details is an "Activity" sidebar with tabs for "New Event", "New Task", "Log a Call", and "Email".

3. The account team field of the account will now be blank.

The screenshot shows the Salesforce Sales tab interface. An account record for "Premium Learnings" is displayed. In the "Account Team" section, there is a button labeled "Add Default Team". The "Customers12" section also has a "New" button. The background features a map-like pattern.

Example 27

When we create the Opportunity with the Probability=20, then the opportunity owner will be automatically added to the Account Team of the associated account for that Opportunity.

```

trigger UpdateATMwithOwneronOptyCreate on Opportunity
(after insert,after update) {

List<AccountShare> list_share= new List<AccountShare>();
List<AccountTeamMember> list_atm=new
                                List<AccountTeamMember>();
for(Opportunity opp:Trigger.New)
{
    if(opp.Probability==20)
    {
        AccountTeamMember atm=new AccountTeamMember();
        atm.accountid=opp.accountid;
        atm.teamMemberRole='Account Manager';
        atm.UserId=opp.Ownerid;
        AccountShare share = new AccountShare();
        share.AccountId=opp.Accountid;
        share.UserOrGroupId = opp.OwnerId;
        share.AccountAccessLevel='Read/Write';
        share.OpportunityAccessLevel = 'Read Only';
        share.CaseAccessLevel='Read Only';
        list_atm.add(atm);
        list_share.add(share);
    }
}
if(list_atm!=null)
    insert list_atm;

if(list_share!=null && list_share.size()>0)
    List<Database.saveresult> sr=
        Database.insert(list_share,false);
}

```

Test Case:

1. Create an account named premium learnings and save it

The screenshot shows the Salesforce Account page for 'Premium Learnings'. The page has a header with a cloud icon, navigation links like Sales, Home, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, and More. Below the header, there's a search bar and a toolbar with Follow, New Contact, New Case, and New Note buttons. The main content area shows the account details for 'Premium Learnings', which is owned by Sanjog Kawade. The 'Details' tab is selected, showing fields for Account Owner, Account Name, Parent Account, Account Number, Account Site, Type, Industry, Annual Revenue, Premium, and out of business status. To the right, there's a Chatter sidebar with tabs for Activity and Chatter, showing a new task button and a message box. A large grey triangle graphic is overlaid on the bottom left of the page.

2. Now go to the related lists and check the account team is empty.

The screenshot shows the Salesforce Account page for 'Premium Learnings' with the Related Lists section visible. The lists include Notes & Attachments (0), Partners (0), consultants (0), customers (0), Assets (0), Account Team (0), and Customers12 (0). Each list has a 'New' button. The 'Account Team (0)' list has buttons for 'Add Default Team' and 'Add Team Members'. A large grey triangle graphic is overlaid on the bottom left of the page.

3. Now go to the related list of the same account, find opportunities, and create a new opportunity (which will be related to this account)

Make sure the probability is 20. Click save

The screenshot shows the Dynamics 365 Sales interface. At the top, there's a navigation bar with links like Sales, Home, Book Your Show, Premium Cab Bookings App, Project Management System, Society Management, Premium Hotels, Premium Hotels and Suites, Opportunities, Leads, Tasks, Files, Accounts, More, and a search bar. A blue banner at the top right indicates "Opportunity 'Premium' was created." On the left, there are related lists for Contacts (1), Opportunities (1), Cases (0), and Notes & Attachments (0). The main area shows a task list with a message: "No activities to show. Get started by sending an email, scheduling a task, and more." The right side has a sidebar with filters, a refresh button, and links for Open All and View All.

4. Now refresh the related lists page of the account and check account team in related lists. It will be populated with opportunity owner of the opportunity record we created

This screenshot shows the same Dynamics 365 Sales interface, but now the Account Team section in the related lists is populated. It shows one entry for "Sanjeet Kawade" with the role "Account Manager". Other sections like Partners, consultants, customers, and Assets remain at zero. The rest of the interface is identical to the previous screenshot.

Example 28

Trigger to update parent records field when child record is updated | update parent field from child using apex trigger

CheckboxStatus

Create two custom objects, both are associated with a lookup relationship with each other

- 1) Custom Object:- Consultant__c custom field:- Status__c (Checkbox boolean type)
- 2) Custom Object:- Leave_Requests__c custom field:- Status__c (Checkbox boolean type)

If the status field of child object marked true then should be status field of parent object automatic marked as true.

```
trigger CheckboxStatus on Leave_Request__c(before Insert,
before update, after insert, after
update, before delete, after delete)
{
    if(trigger.isBefore){
        //system.debug('I am inside before event');
    }
    else if(trigger.isAfter){
        //system.debug('I am inside after event');

        if(trigger.isUpdate){
            for(Leave_Request__c myStatus: Trigger.new){
                Consultant__c getParentObj = [SELECT Id,
                                                Status__c FROM Consultant__c WHERE Id =
                                                :myStatus.Consultant__c];
                Leave_Request__c getChildObj = [SELECT Id,
                                                Status__c, Consultant__r.Status__c FROM
                                                Leave_Request__c WHERE Id = :myStatus.Id];
                getParentObj.Status__c= getChildObj.Status__c;
                insert getParentObj;
            }
        }
    }
}
```

```

if(trigger.isInsert && (trigger.isAfter)){
    for(Leave_Request__c myStatus: Trigger.new){
        Consultant__c getParentObj = [SELECT Id,
            Status__c FROM Consultant__c WHERE Id =
            :myStatus.Consultant__c ];
        Leave_Request__c getChildObj = [SELECT Id,
            Status__c, Consultant__r.Status__c FROM
            Leave_Request__c WHERE Id = :myStatus.Id];
        getParentObj.Status__c= getChildObj.Status__c;
        update getParentObj;
    }
}

if(Trigger.isDelete && (Trigger.isBefore)){
    for(Leave_Request__c myStatus: Trigger.old){
        Consultant__c getParentObj = [SELECT Id, Status__c
            FROM Consultant__c WHERE Id =
            :myStatus.Consultant__c ];
        Leave_Request__c getChildObj = [SELECT Id,
            Status__c, Consultant__r.Status__c FROM
            Leave_Request__c WHERE Id = :myStatus.Id];
        delete getParentObj;
    }
}
}

```

Test Case:

1. Create new record of Consultant and (Keep the Status checkbox unchecked).

The screenshot shows a Salesforce interface for a 'Consultant' record named 'Ankit'. The 'Details' tab is selected, displaying fields like Consultant Name (Ankit), Type, Email, Account (Mr Ashish), and Status (unchecked). The 'Owner' field shows 'Amruta Lokhande'. The 'Created By' and 'Last Modified By' fields both show 'Amruta Lokhande' on 7/6/2022, 3:13 PM. To the right, the 'Activity' sidebar is open, showing tabs for New Event, New Task, Log a Call, and Email. A sub-section for 'Upcoming & Overdue' activities is visible, stating 'No activities to show.' and 'Get started by sending an email, scheduling a task, and more.'

2. Create Child Leave_Request for the same consultant (check the Status checkbox).

The screenshot shows a 'New Leave Request' dialog box. In the 'Information' section, there are three fields: 'Leave Request Name' (Diwali Leaves), 'Consultant' (Ankit), and 'Status' (checkbox checked). At the bottom of the dialog are 'Cancel', 'Save & New', and 'Save' buttons.

The screenshot shows the Salesforce Leave Request page for a leave request named "Diwali Leaves". The "Details" tab is selected. The "Status" field has a checked checkbox. The "Activity" sidebar is open, showing tabs for "New Event", "New Task", "Log a Call", and "Email". A button to "Set up an event..." is present, along with a "Filters" dropdown and "Refresh" buttons.

3. Check the Status checkbox of parent consultant will automatically become checked.

The screenshot shows the Salesforce Consultant page for a consultant named "Ankit". The "Details" tab is selected. The "Status" field has a checked checkbox. The "Activity" sidebar is open, showing tabs for "New Event", "New Task", "Log a Call", and "Email". A button to "Set up an event..." is present, along with a "Filters" dropdown and "Refresh" buttons.



Example 29

Write a trigger on Account to Prevent the user to create duplicate Account based on Phone if Phone number is already exist in Salesforce

Trigger :

checkDuplicationPhoneAccTrigger.apxt

```

trigger checkDuplicationPhoneAccTrigger on Account
(before insert,after insert)
{
    if(trigger.isBefore){
        system.debug('trigger before trigger');

        if(trigger.isInsert){
            set<String> accPhone=new set<String>();
            for(Account a:Trigger.new){
                accPhone.add(a.Phone);
                System.debug('Account phone number is '
                            + accPhone);
            }
            set<String> phoneAccData=new set<String>();
            for(Account a:[select Id, Name, Phone from Account
                           where Phone in:accPhone])
            {
                phoneAccData.add(a.Phone);
            }
            for(Account a:Trigger.new){
                if(phoneAccData.contains(a.Phone)){
                    a.addError('Do not allow insert duplicate
                               phone number');
                }
            }
        }
    }
    else if(trigger.isAfter){
        system.debug('trigger after trigger');
    }
}

```



Test Case:

1. Create new Account record .

The screenshot shows the Salesforce interface for creating a new account. The top navigation bar includes Sales, Home, Opportunities, Leads, Tasks, Files, Accounts, Contacts, Campaigns, Dashboards, Reports, Chatter, Groups, Calendar, People, Consultants, and More. The main page title is 'Account New Account'. The 'Details' tab is selected, showing fields for Type (Phone: (987) 654-3210), Account Owner (Amruta Lokhande), Account Site, Industry, Rating (Hot), Phone, Fax, Website, Ticker Symbol, Ownership, Employees (50), SIC Code, and Contact Last updated. The 'Activity' sidebar displays a 'New Task' section with a 'Create a task...' input field and an 'Add' button, along with filters and a 'Upcoming & Overdue' section indicating no activities.

2. Try to create another Account record with same Phone. And see it will display an error message as shown in below screenshot.

The screenshot shows the 'New Account' creation page again. The 'Phone' field is now populated with '9876543210'. A red error message box appears in the center of the page with the text 'We hit a snag.' and a note: 'Review the errors on this page.' followed by a bulleted list: '• Do not allow insert duplicate phone number'. The 'Save' and 'Save & New' buttons are visible at the bottom of the form.

Example 30

Trigger to check duplicate name to custom object in Salesforce | how to prevent duplicate records based on multiple fields through apex trigger in Salesforce

Trigger : childObjTriggerHandler.apxt

Create Custom Objects.

1.Parent object: Project__c

2.Child Object: Employee__c, Field: Employee_Name__c

Create look relationship between them.

```
trigger childObjTriggerHandler on Employee__c
(before insert, after insert)
{
    if(trigger.isBefore){
        if(trigger.isInsert){
            system.debug('I am inside before insert');
            childObjHandler.childDuplicate(trigger.new);
        }
    }
    else if(trigger.isAfter){
        if(trigger.isInsert){
            //system.debug('I am inside after insert');
            system.debug('Record inserted successfully');
        }
    }
}
```



Class: :ChildObjHandler

```

public class ChildObjHandler {

    public static void childDuplicate
        (List<Employee__c> dupChildRec)
    {

        Set<String> empExtStr = new Set<String>();
        List<Employee__c> empList = [Select Id, Name,
                                      EmployeeName__c, Project__c From
                                      Employee__c where EmployeeName__c !=null];

        for(Employee__c empStr:empList){
            empExtStr.add(empStr.EmployeeName__c)
        }

        for(Employee__c childTr:dupChildRec){
            if(empExtStr.contains(childTr.EmployeeName__c))
            {
                childTr.EmployeeName__c.addError
                    ('Do not allow duplicate');
            }
        }
    }
}

```



Test Case:

1. Create Child Employee record .

The screenshot shows the Salesforce Employee details page for an employee named Amruta. The 'Details' tab is active. Key fields visible include Employee Name (Amruta), Employee ID (E-0010), Email, Phone, Designation, Skills, Project (Airline Reservation System), Requirement, and EmployeeName. The Owner is listed as Amruta Lokhande. The Activity sidebar indicates no upcoming or overdue activities.

2. Try to create new Employee record with same Name it will give you an error as shown in below screenshot.

The screenshot shows the 'New Employee' creation dialog. The Employee Name field is highlighted in yellow. A red error message box at the bottom left says 'We hit a snag.' with the sub-instruction 'Review the following fields: EmployeeName'. The EmployeeName field is also highlighted in red. Other fields like Email, Phone, Designation, Skills, and Project are present but not highlighted.



Example 31

Create rollup summary using Apex trigger on custom

Trigger: childObjTriggerRollUp

Create a custom field on parent Object name as childRollUp_c (Number Type). Create a lookup relationship field on parent Object

```
trigger childObjTriggerRollUp on Contact (before insert,
before update, before delete, after insert, after update,
after delete, after undelete) {

List<Contact> childObjList = new List<Contact>();
Set<Id> accSetId = new Set<Id>();
if(trigger.isBefore){
    system.debug('trigger before#');
}
else if(trigger.isAfter){
    system.debug('trigger after#');
    if(trigger.isDelete){
        childObjList = trigger.old;
    }
}
else{
    childObjList = trigger.new;
}
List<Account> updateParent = new List<Account>();

for(Contact childObjNewList:childObjList){
    if(childObjNewList.AccountId != null){
        accSetId.add(childObjNewList.AccountId);
        if(trigger.isUpdate){
            Contact childObjOld =
                trigger.oldMap.get(childObjNewList.Id);
            if(childObjOld != null)
                accSetId.add(childObjOld.AccountId);
        }
    }
}
if(accSetId.size() > 0){
    Account[] accList = [SELECT Id, Name, childRollUp_c FROM Account WHERE Id IN :accSetId];
    for(Account acc: accList){
        acc.childRollUp_c = acc.childObjList.size();
    }
    updateParent.addAll(accList);
}
}
```

```
system.debug('childObjOld ' + childObjOld);
if(childObjOld.AccountId !=  
    childObjNewList.AccountId)  
{  
    accSetId.add(childObjOld.AccountId);  
}  
else if(childObjOld.AccountId == null)  
{  
    accSetId.add(childObjNewList.AccountId);  
}  
}  
}  
  
for(Account parentSetObj:[Select Id, Name,ChildRollup__c  
From Account Where Id IN:accSetId])  
{  
    List<Contact> childSize = parentSetObj.Contact__r;  
    parentSetObj.ChildRollup__c=childSize.size();  
    updateParent.add(parentSetObj);  
}  
update updateParent;  
}
```





Test Case:

1. Create new Account Record.

Account Test1

Type: Prospect | Phone: (988) 775-4434 | Website: | Account Owner: Amruta Lokhande | Account Site: | Industry: Banking

Related Details News

Account Owner	Rating
Amruta Lokhande	Hot
Account Name	Phone
Test1	(988) 775-4434
Parent Account	Fax
Account Number	Website
Account Site	Ticker Symbol
Type	Ownership
Prospect	Employees
Industry	SIC Code
Banking	
Annual Revenue	
Support Level	
UpdateContact	

Activity Chatter

- New Task
- Log a Call
- New Event
- Email

Create a task... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

2. Create three child Contact records of same Account.

Contact Con 1

Contact Owner: Amruta Lokhande | Name: Con 1 | Account Name: Test1

Related Details News

Contact Owner	Phone
Amruta Lokhande	(988) 775-4434
Name	Home Phone
Con 1	Mobile
Account Name	Other Phone
Test1	Fax
Title	Email
Department	Assistant
Birthdate	Ext. Phone
Reports To	
Lead Source	
Last Modified Date	
Account Website	

Activity Chatter

- New Task
- Log a Call
- New Event
- Email

Create a task... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

Contact
con2

Title: Test1 Account Name: Test1 Phone (2): (988) 775-4434 Email: [redacted] Contact Owner: Amruta Lokhande

Related **Details** News

Contact Owner	Phone
Amruta Lokhande	(988) 775-4434
Name	Home Phone
con2	
Account Name	Mobile
Test1	
Title	Other Phone
Department	Fax
Birthdate	Email
Reports To	Assistant
Lead Source	Asst. Phone

Activity Chatter

New Task Log a Call New Event Email

Create a task... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

Contact
con3

Title: Test1 Account Name: Test1 Phone (2): (988) 775-4434 Email: [redacted] Contact Owner: Amruta Lokhande

Related **Details** News

Contact Owner	Phone
Amruta Lokhande	(988) 775-4434
Name	Home Phone
con3	
Account Name	Mobile
Test1	
Title	Other Phone
Department	Fax
Birthdate	Email
Reports To	Assistant
Lead Source	Asst. Phone

Activity Chatter

New Task Log a Call New Event Email

Create a task... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.



2. On Account record check the childRollup field. It shows the number of child contacts that account has. As shown in below screenshot

The screenshot shows the Salesforce Account detail page for an account named 'Test1'. The 'ChildRollup' field is highlighted with a red box and contains the value '3'. The page includes a navigation bar with links like Sales, Home, Opportunities, Quotes, Leads, Tasks, Files, Accounts, Contacts, Campaigns, Dashboards, Reports, Chatter, Groups, and More. The main content area displays various account details such as Name, Phone, Fax, Website, Ticker Symbol, Ownership, Employees, SIC Code, Type (Prospect), Industry (Banking), Annual Revenue, Support level, updateContact, Premium status (unchecked), and Amount. To the right, there is a sidebar with sections for Upcoming & Overdue tasks (empty) and Past activity (empty).



Example 32

Write a trigger on Contact and Add First Name and Last Name of Contact to Associated Account Custom Field Whenever Contact record is Inserted or Updated.

Trigger: ContactAccountTrigger

```
trigger ContactAccountTrigger on Contact (before insert,
before update, after insert, after update) {

List<Contact> conList =new List<Contact>();
Set<Id> setid = new Set<Id>();

if(trigger.isBefore){
    system.debug('trigger before event');
    conList = trigger.new;
}

else if(trigger.isAfter){
    conList=trigger.new;
    for(Contact con:conList){
        setid.add(con.AccountId);
    }
    system.debug('setid ' + setid);

List<Account> accList = [Select Id, Name, updateContact__c
                           From Account Where Id=:setid];
}
```

```
if(trigger.isInsert){  
    for(Contact c1:trigger.new){  
        for(Account a1:accList){  
            a1.updateContact__c= c1.FirstName + ''  
                + c1.LastName;  
            update a1;  
        }  
    }  
  
    else if(trigger.isUpdate){  
        for(Contact c2:trigger.new){  
            for(Account a2:accList){  
                a2.updateContact__c= c2.FirstName +  
                    ' ' + c2.LastName;  
                update a2;  
            }  
        }  
    }  
}
```



Test Case:

1. Create new Account record.

The screenshot shows the Salesforce Accounts page. At the top, there's a navigation bar with links like Sales, Home, Opportunities, Quotes, Leads, Tasks, Files, Accounts (which is the active tab), Contacts, Campaigns, Dashboards, Reports, Chatter, Groups, and More. Below the navigation is a search bar and some quick action buttons: + Follow, New Contact, New Case, and New Note. The main area displays an account record for "Pooja". The account details include Type (Account), Phone ((123) 456-7890), Website, Account Owner (Amruta Lokhande), Account Site, and Industry (Banking). On the left, there are tabs for Related, Details (which is selected), and News. On the right, there's an Activity sidebar with sections for New Task, Log a Call, New Event, Email, and a button to Add. It also shows Chatter activity and filters for Upcoming & Overdue tasks.

2. Create new child contact record with contact Last name and First name..

The screenshot shows the "New Contact" dialog box. It has a header "New Contact" and a section titled "Contact Information". The form fields include: Contact Owner (Amruta Lokhande), Name (highlighted in yellow), Salutation (None), First Name (Ashish), Last Name (Lokhande), Account Name (Pooja), Title, Department, Phone (1234567890), Home Phone, Mobile, Other Phone, Fax, and Birthdate. At the bottom are buttons for Cancel, Save & New, and Save.



3: Check Account custom field that is Updated_Contact , Contact first name and Last name is associated with it.

The screenshot shows the Salesforce Account detail page for an account named 'Pooja'. The page is divided into two main sections: 'Details' and 'Activity'.

Details Section:

- Account Owner:** Amrita Lokhande
- Account Name:** Pooja
- Parent Account:** [Blank]
- Account Number:** [Blank]
- Account Site:** [Blank]
- Type:** Banking
- Industry:** Banking
- Annual Revenue:** [Blank]
- Support Level:** [Blank]
- updateContact:** Ashish Lokhande
- Premium:** [checkbox]
- Amount:** [Blank]

Activity Section:

- Activity:** Chatter
- Buttons:** New Task, Log a Call, New Event, Email
- Text Input:** Create a task... (with an 'Add' button)
- Filter:** Filters: All time • All activities • All types
- Section:** Upcoming & Ongoing
- Text:** No activities to show. Get started by sending an email, scheduling a task, and more.
- Text:** No past activity. Past meetings and tasks marked as done show up here.



Example 33

trigger on contact which will create Account record whenever contact is created without an account.

Trigger: ContactCustomTriggerExample.apxt

```
trigger ContactCustomTriggerExample on Contact(after insert)
{
    List<Account> accListToInsert = new List<Account>();
    for(Contact con : Trigger.New) {
        //check if account is null on contact
        if(con.AccountId == null ) {
            Account acc = new Account();
            //Add all required field on Account
            acc.Name = con.LastName;
            acc.Phone = con.Phone;
            accListToInsert.add(acc);
        }
    }
    if(!accListToInsert.isEmpty()){
        insert accListToInsert;
    }
}
```



Test Case:

1. Create new contact record without parent account.

2. Check in All Accounts new account record will create automatically with name same as contact last name .

Type	Phone	Website	Account Owner	Account Site	Industry
Account Owner	Amruta Lokhande		Amruta Lokhande		
Account Name	Test Contact				
Parent Account					
Account Number					
Account Site					
Type					
Industry					
Annual Revenue					



Example 34

Write a trigger to update parent account phone number whenever the contact phone number is updated using trigger handler and helper class in Salesforce

Trigger: contactTrigger

```
trigger contactTrigger on Contact (before insert, before update, before delete, after insert, after update, after delete, after undelete)
{
    if(trigger.isBefore ){
        system.debug('I am before trigger ');
    }
    else if(trigger.isAfter){
        system.debug('I am after trigger ');
        if(trigger.isUpdate){
            contactTriggerHandler.afterUpdateHelper(trigger.new);
        }
    }
}
```



Class :

```

public class contactTriggerHandler {
    public static void afterUpdateHelper(List<Contact> conList)
    {

        Set<Id> setId = new Set<Id>();

        for(Contact con:conList){

            setId.add(con.AccountId);
        }

        system.debug('setId ' + setId);

        List<Account> accItemList = [Select Id, Name, Phone,
                                      (Select Id, FirstName, LastName, Phone,
                                       AccountId From Contacts) From Account
                                      Where Id IN:setId];

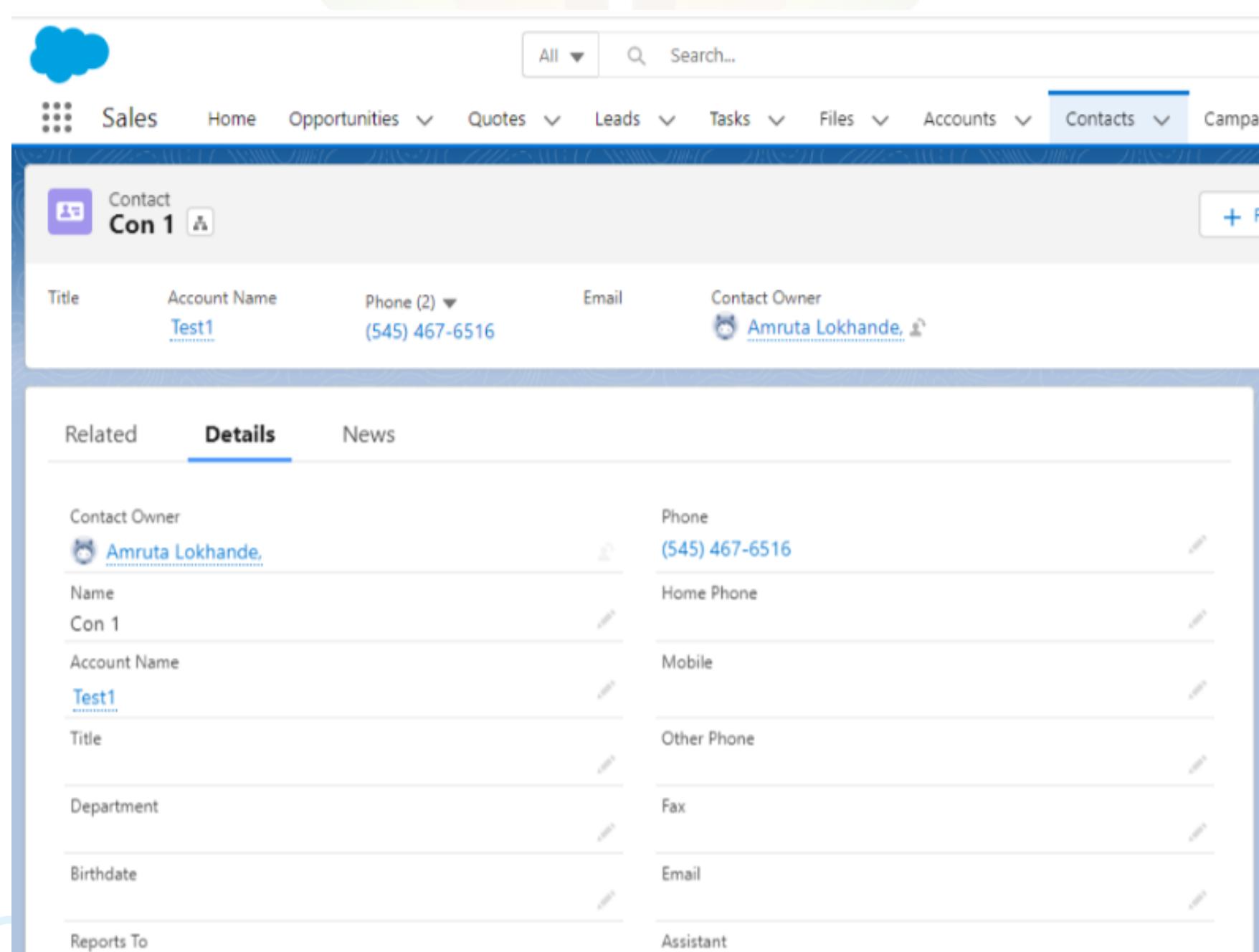
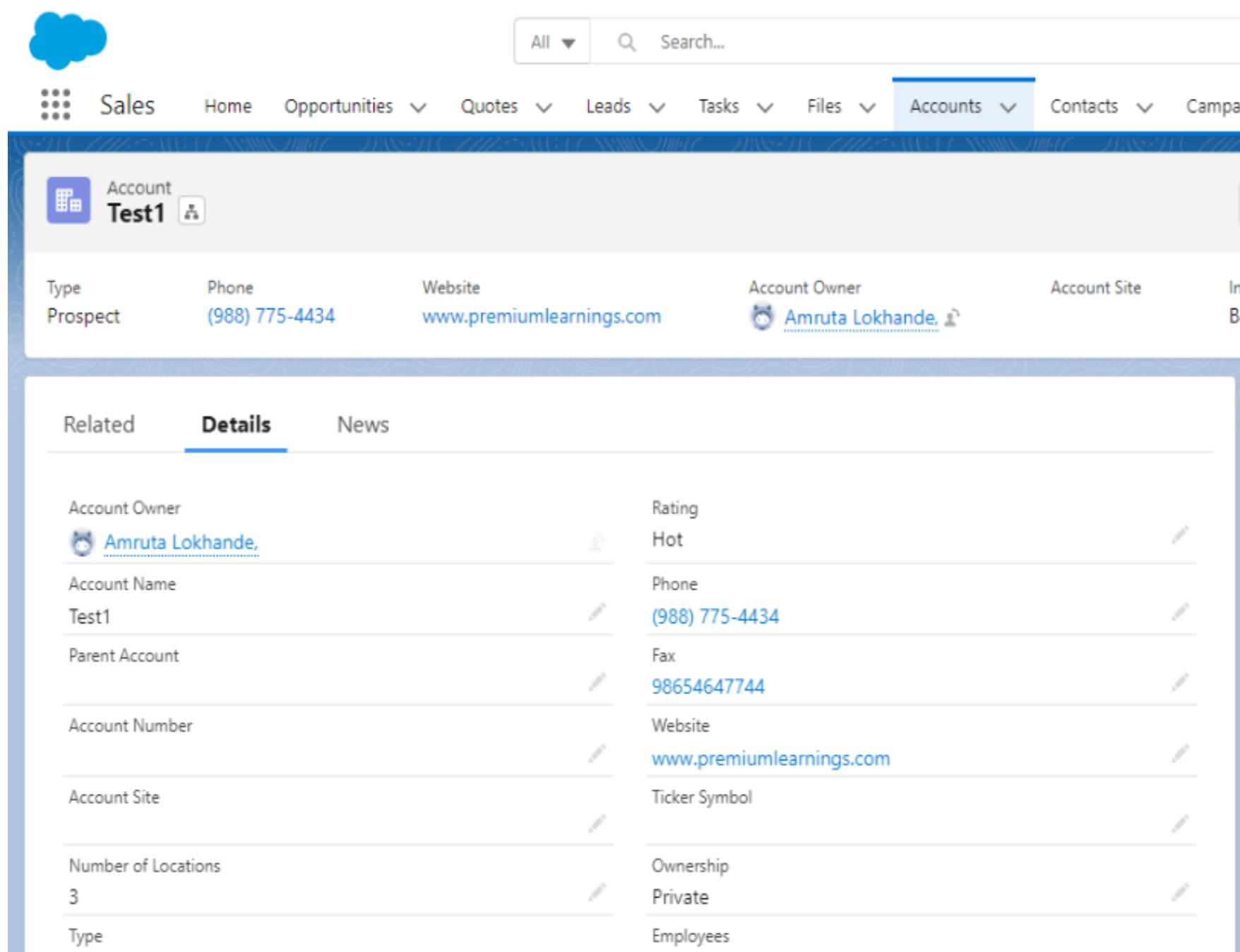
        for(Account a1:accItemList){
            for(Contact c1:a1.Contacts){
                if(c1.Phone !=null){
                    a1.Phone = c1.Phone;
                    update accItemList;
                }
            }
        }
    }
}

```



Test Case:

Create Account and contact record with phone number as shown below.





2: Now update the phone field of contact and you can see the parent account phone number will update whenever the contact phone number is updated

Edit Con 1

Contact Owner Amruta Lokhande,	Phone 9876543210
*Name Salutation --None--	Home Phone
First Name First Name	
*Last Name Con 1	
Account Name Test1	Mobile
Title Title	Other Phone
Department Department	<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>

Now check phone field of parent account.

Sales Home Opportunities Quotes Leads Tasks Files Accounts Contacts Campaigns

Contact Con 1

Title	Account Name Test1	Phone (2) (987) 654-3210	Email	Contact Owner Amruta Lokhande,
-------	-----------------------	-----------------------------	-------	-----------------------------------

Related Details News

Contact Owner Amruta Lokhande.	Phone (987) 654-3210
Name Con 1	Home Phone
Account Name Test1	Mobile
Title	Other Phone
Department	Fax
Birthdate	Email
Reports To	Assistant

Example 35

When ever the Account is created with Industry as Banking then create a contact for account, Contact Lastname as Account name and contact phone as account phone.

Trigger: CreateAccountContact.apxt

Object : Account

Trigger: After Insert

```
trigger CreateAccountContact on Account (after insert)
{
    list<contact> contacts = new list<contact>();
    for(account acc : trigger.new){
        if(acc.Industry == 'Banking'){
            contact con = new contact();
            con.LastName = acc.name;
            con.Phone = acc.Phone;
            con.AccountId = acc.Id;
            contacts.add(con);
        }
    }
    insert contacts;
}
```



Test Case:

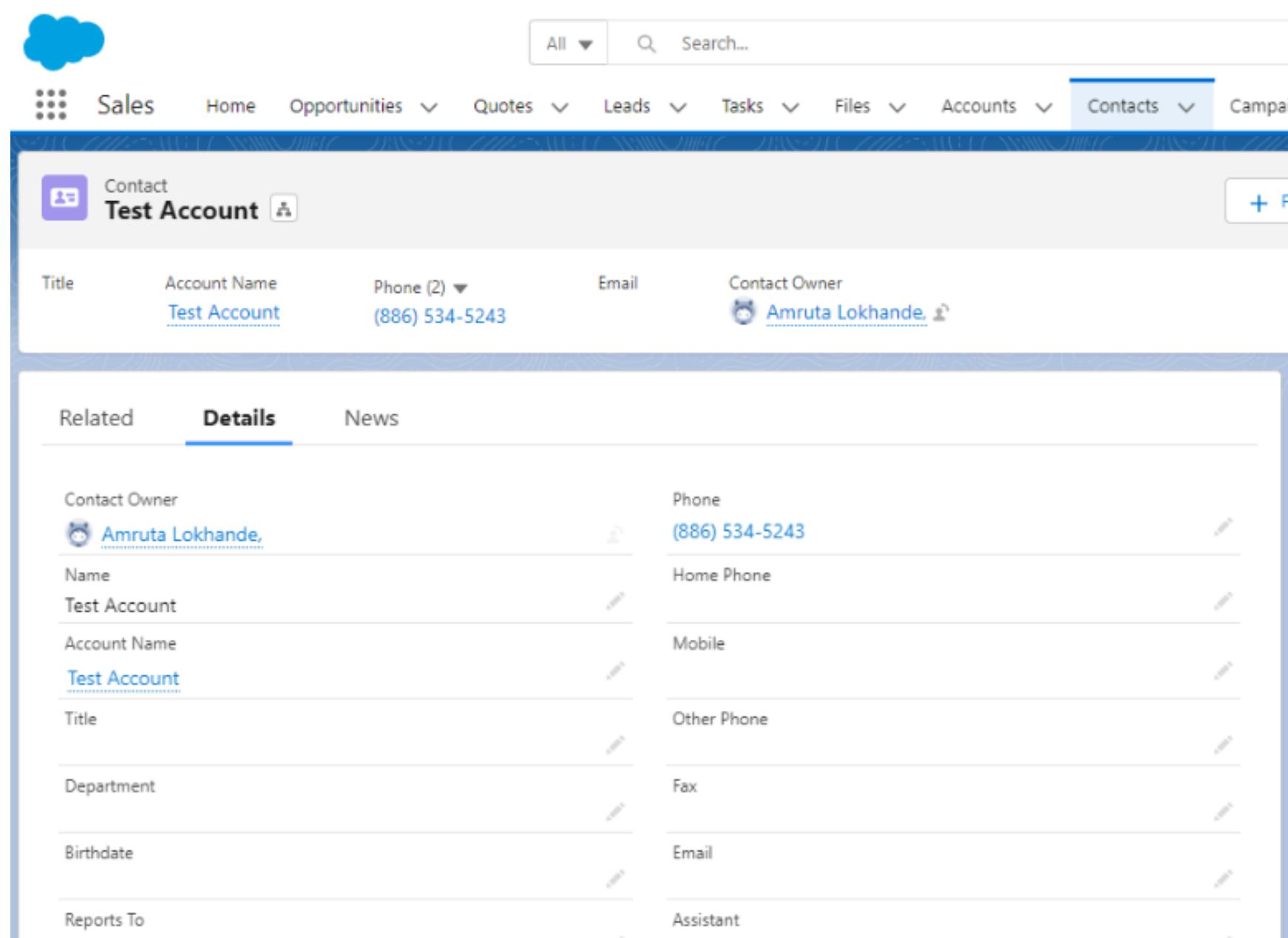
- Create Account record and industry must be Banking as shown in below screenshot.

The screenshot shows the Salesforce Account creation interface. The 'Industry' field is highlighted in yellow and set to 'Banking'. Other fields include Account Name (Test Account), Phone (8865345243), Fax (9976543456), Website (www.premiumlearnings.com), Number of Locations (2), Type (Customer - Direct), Ownership (Public), and Annual Revenue (0). Buttons at the bottom are Cancel, Save & New, and Save.

- Check the new child Contact record will create automatically with same name as like Account Last name and with same Phone.

The screenshot shows the Salesforce Account detail page for 'Test Account'. The account details include Type (Customer - Direct), Phone ((886) 534-5243), Website (www.premiumlearnings.com), Account Owner (Amruta Lokhande), and Account Site. The 'Related' tab is selected, showing one contact named 'Test Account' with the same phone number. There are also sections for Opportunities (0) and View All.

Contact Last name as Account name and contact phone as account phone.



Example 36

Apex Trigger on Contact to Create Account Automatically and map to related account Id to Contact Whenever New Contact is Created in Salesforce | trigger to create account automatically whenever contact is created in salesforce.

Trigger: CreateAutoAccountFromContact

```
trigger CreateAutoAccountFromContact on Contact (before insert, after insert) {
    if(trigger.isBefore){
        system.debug('trigger before event');
        if(trigger.isInsert){
            List<Contact> listOfContact=new List<Contact>();

            for (Contact conObj : trigger.new)
            {
                if(String.isBlank(conObj.accountid))
                {
                    listofContact.add(conObj);
                }
            }

            system.debug('listOfContact_1 '+ listOfContact);

            if (listOfContact.size() > 0)
            {
                List<Account> createNewAcc=new List<Account>();
                Map<String, Contact> conNameKeys = new
                    Map<String, Contact>();
            }
        }
    }
}
```

```
for (Contact con : listOfContact)
{
    String accountName = con.firstname + ' ' +
                           con.lastname;
    conNameKeys.put(accountName,con);
    Account accObj = new Account();
    accObj.Name = accountName;
    accObj.Phone= con.MobilePhone;
    createNewAcc.add(accObj);
}
Insert createNewAcc;
for (Account acc : createNewAcc)
{
    System.debug('mapContainsKey ' +
                 conNameKeys.containsKey(acc.Name));

    if (conNameKeys.containsKey(acc.Name))
    {
        conNameKeys.get(acc.Name).accountId = acc.Id;
    }
}
}
else if(trigger.isAfter){
    system.debug('trigger after event');
}
```





Test Case:

1. Create new Contact record without Account as shown in below screenshot.

New Contact

Contact Information

Contact Owner: Amruta Lokhande

*Name: Test

Salutation: --None--

First Name: First Name

*Last Name: Test

Account Name: Search Accounts...

Mobile:

Title:

Other Phone:

Department:

Buttons: Cancel, Save & New, Save

2. Check related account record will create automatically whenever contact is created. k the new child Contact record will create automatically with same name as like Account Last name and with same Phone.

Contact Test

Title: null

Account Name: null

Phone (2): (987) 653-4654

Email:

Contact Owner: Amruta Lokhande

Related Details News

Contact Owner: Amruta Lokhande

Name: Test

Account Name: null

Title:

Department:

Birthdate:

Reports To:

Phone: (987) 653-4654

Home Phone:

Mobile:

Other Phone:

Fax:

Email:

Assistant:

Activity Chatter

New Task Log a Call New Event Email

Create a task... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up

Example 37

The number of contacts which are equal to the number which we will enter in the Number of Locations field on the Account Object.

Trigger: CreateNoOfContacts

Create a custom field on parent Object name as NumberofLocations__c (Number Type).

```
trigger CreateNoOfContacts on Account (after insert)
{
    list<contact> listContact = new list<contact>();
    map<id,decimal> mapAcc=new map<id,decimal>();

    for(Account acc:trigger.new){
        mapAcc.put(acc.id,acc.NumberofLocations__c);
    }
    if(mapAcc.size()>0 && mapAcc!=null){
        for(Id accId:mapAcc.keySet()){
            for(integer i=0;i<mapAcc.get(accId);i++){
                contact newContact=new contact();
                newContact.accountid=accId;
                newContact.lastname='contact'+i;
                listContact.add(newContact);
            }
        }
    }
    if(listContact.size()>0 && listContact!=null)
        insert listContact;
}
```



Test Case:

1. Create Account record and must enter Number of locations as shown in below screenshot.

Edit Test1

Account Owner Amruta Lokhande,	Rating Hot
*Account Name Test1	Phone 9887754434
Parent Account Search Accounts...	Fax 98654647744
Account Number	Website www.premiumlearnings.com
Account Site	Ticker Symbol
Type Prospect	Ownership Private
Industry Banking	Employees
Annual Revenue \$6,760,000	SIC Code
Support level	
<input type="button" value="updateContact"/> <input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>	

Billing City	Billing State/Province	Shipping City	Shipping State/Province
Billing Zip/Postal Code	Billing Country	Shipping Zip/Postal Code	Shipping Country
Customer Priority High	SLA Gold	SLA Expiration Date	SLA Serial Number
Number of Locations 3	Upsell Opportunity Maybe	Description	
Active --None--		Created By Amruta Lokhande,, 7/6/2022, 4:34 AM	
		Last Modified By Amruta Lokhande,, 7/7/2022, 4:08 AM	
Billing			
<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>			

2. Check the number of contacts will create which are equal to the number which we will enter in the Number of Locations field on the Account Object.

The screenshot shows the Salesforce Account page for an account named 'Test1'. The account details include:

- Type: Prospect
- Phone: (988) 775-4434
- Website: www.premiumlearnings.com
- Account Owner: Amruta Lokhande
- Account Site: Not specified
- Industry: Banking

The 'Related' section displays:

- Contacts (3)**: Shows three contacts: Con 1, con2, and con3, each with a title, email, and phone number.
- Opportunities (0)**: Shows 0 opportunities.

The 'Activity' section shows:

- New Task, Log a Call, New Event, Email buttons.
- A 'Create a task...' input field and an 'Add' button.
- Filters: All time • All activities • All types.
- No activities to show. Get started by sending an email, scheduling a task, and more.
- No past activity. Past meetings and tasks marked as done show up.



Example 38

Create the object called “Books” and create field “Price”(data type is Currency) under this object. Whenever we enter some amount of money in the Price field and once we click on save button, the value we entered in the Price field is 10% less than the actual price. This is applicable for while both inserting and updating records.

Trigger: DiscountTrigger.apxt

Create custom object.

1)Custom Object:- Book__c custom field:- Price__c (Currency type)

```
trigger DiscountTrigger on Book__c (before insert,  
before update)  
{  
    List<Book__c> books = Trigger.new;  
    PriceDiscount.applyDiscount(books);  
}
```

Class : PriceDiscount.apxc

```
public class PriceDiscount  
{  
    public static void applyDiscount(List<Book__c> books)  
    {  
        for (Book__c b :books)  
        {  
            b.Price__c *= 0.9;  
        }  
    }  
}
```



Test Case:

- Create record of Book object as shown in the below screenshot.

New Book

Information

* Book Name: The Secret

Price: \$100

Owner: Amruta Lokhande

Cancel Save & New Save

- Check the value we entered in the Price field is 10% less than the actual price.

All Search Books and more..

Sales Home Opportunities Quotes Leads Tasks Files Accounts Contacts Campaigns Dashboards Reports Chatter Books More

Book The Secret

New Contact Edit New Opportunity

Related Details

Book Name: The Secret

Owner: Amruta Lokhande

Price: \$90

Created By: Amruta Lokhande, 7/7/2022, 4:06 AM

Last Modified By: Amruta Lokhande, 7/7/2022, 4:06 AM

Activity

New Event New Task Log a Call Email

Set up an event... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

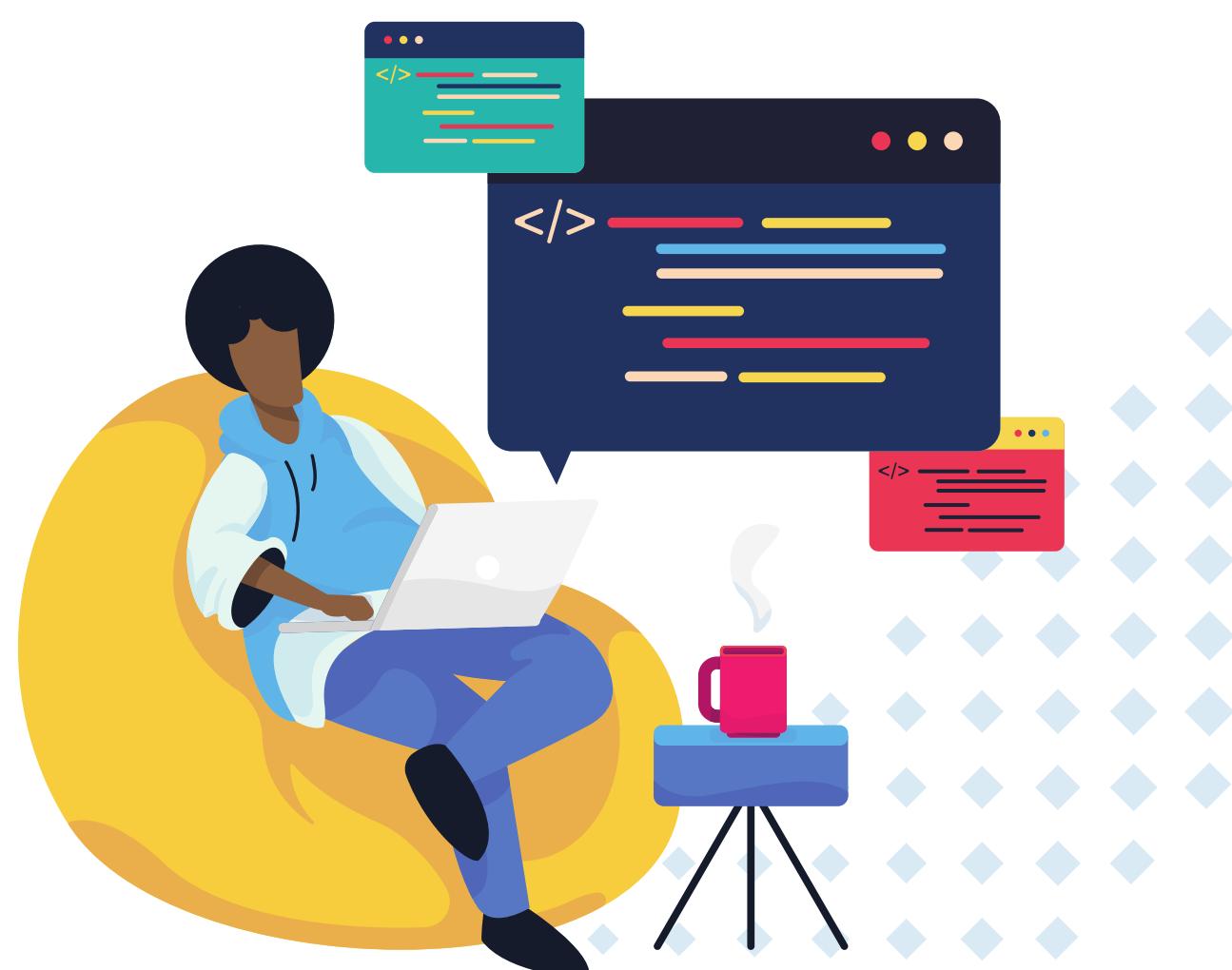
Example 39

Write a trigger whenever Opportunity is deleted the corresponding Account and Contact should be deleted uses of Apex trigger in Salesforce.

Trigger: OpportunityDeleteTrigger

```
trigger OpportunityDeleteTrigger on Opportunity (before delete, after delete){  
    if(trigger.isBefore){  
        system.debug('trigger before trigger');  
    }  
    else if(trigger.isAfter){  
        system.debug('trigger after trigger');  
        if(trigger.isDelete){  
            Map<Id,Opportunity> oppItemMap=new  
            Map<Id,Opportunity>();  
            for(Opportunity opp:Trigger.old){  
                oppItemMap.put(opp.AccountId,opp);  
            }  
  
            List<Account> oppRelListDelete=new List<Account>();  
            List<Account> accObjlist=[select Id,Name, Industry,  
                Website, Type, Phone, Description From  
                Account where Id in:oppItemMap.keySet()];  
  
            for(Account acc:accObjlist){  
                oppRelListDelete.add(acc);  
            }  
        }  
    }  
}
```

```
if(!oppRelListDelete.isEmpty() &&  
    oppRelListDelete.size()>0)  
{  
    delete oppRelListDelete;  
}  
}  
}
```



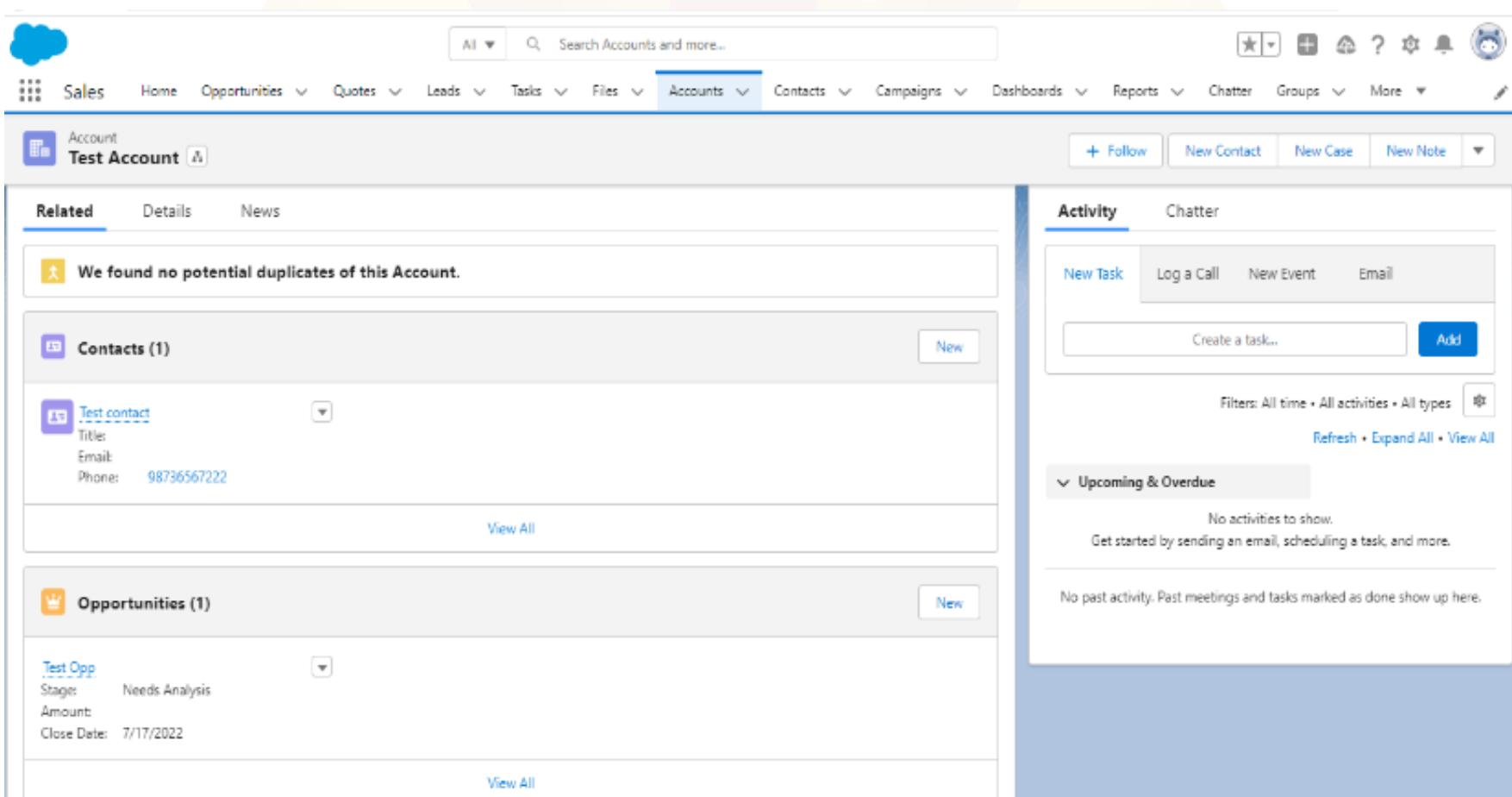
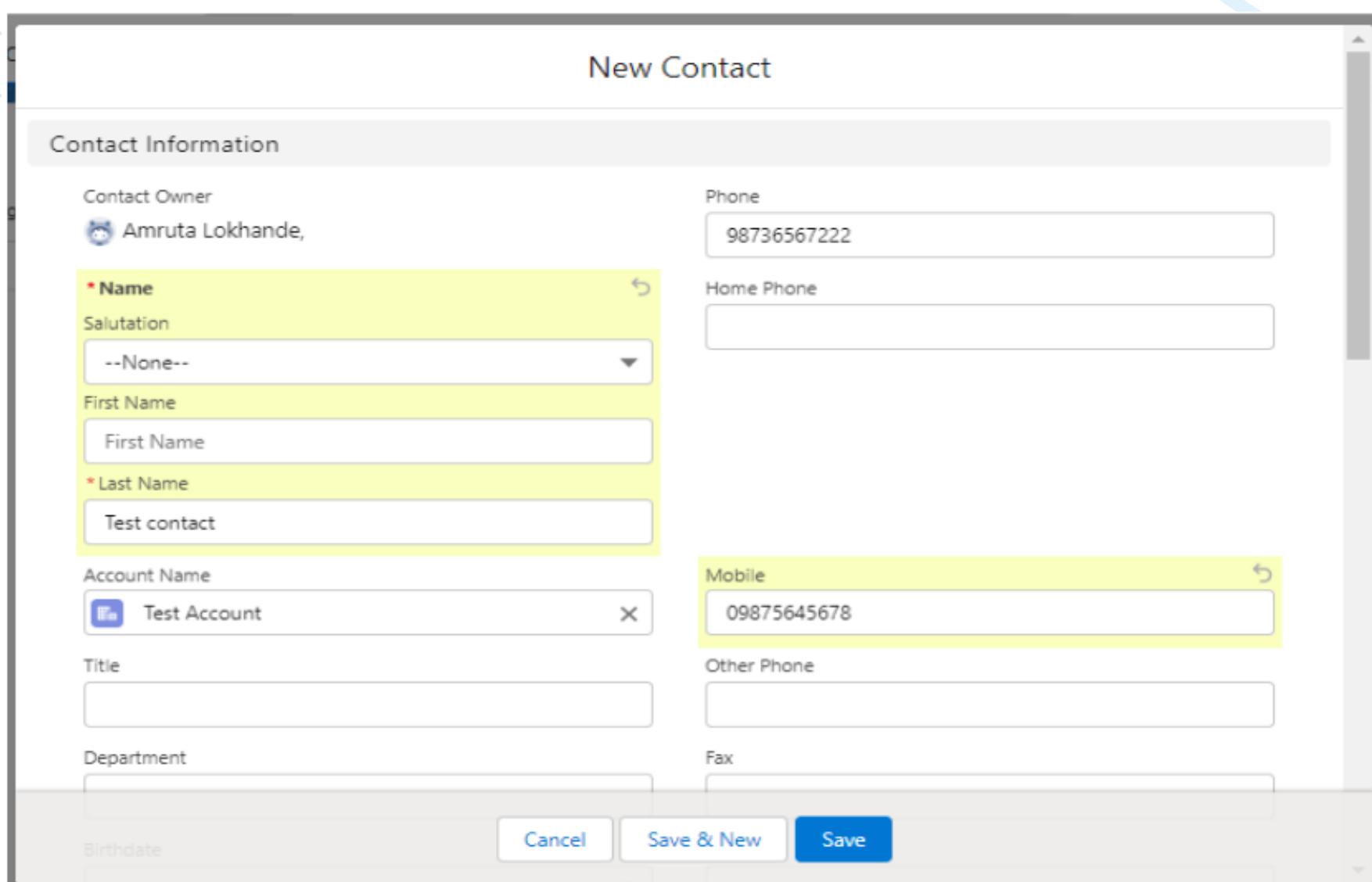
Test Case:

1. Create new Opportunity record as shown in below screenshot.

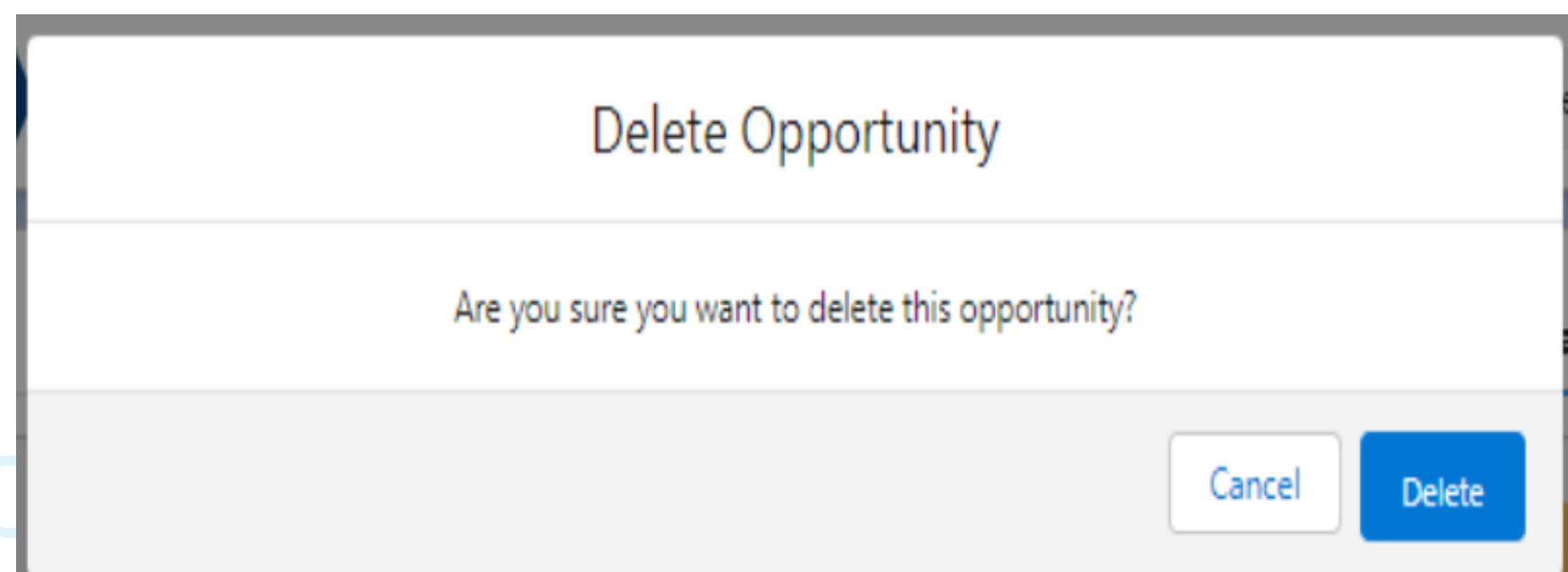
The screenshot shows the 'New Opportunity' form. The 'Opportunity Name' field is highlighted with a yellow background and contains the value 'Test Opp'. Other fields visible include 'Account Name' (Test Account), 'Type' (Existing Customer - Upgrade), 'Lead Source' (Purchased List), 'Status' (Submitted), 'Amount' (empty), 'Close Date' (7/17/2022), 'Next Step' (empty), 'Stage' (Needs Analysis), 'Probability (%)' (20%), and 'Primary Campaign Source' (Search Campaigns...).

2. Create new Account and contact record related to opportunity.

The screenshot shows the 'New Account' form. The 'Account Name' field is highlighted with a yellow background and contains the value 'Test Account'. Other fields visible include 'Rating' (Hot), 'Phone' (98736567222), 'Fax' (empty), 'Website' (www.premiumlearnings.com), 'Ticker Symbol' (empty), 'Ownership' (Private), and 'Employees' (empty). Fields like 'Parent Account' and 'Account Number' are also present.



3. When you delete Opportunity corresponding Account and Contact will be deleted.



Sales Home Opportunities Quotes Leads Tasks Files Accounts Contacts Campaigns Dashboards Reports Chatter Groups More

Recently Viewed

9 Items • Updated a few seconds ago

	Account Name	Account Site	Phone	Account Owner Alias
1	Pooja		1234567890	ALokh
2	Test		(987) 654-3221	ALokh
3	Test1		(988) 775-4434	ALokh
4	Omkar		(987) 654-3210	ALokh
5	New Account			ALokh
6	Test2			ALokh
7	Test1			ALokh
8	null Contact 4			ALokh
9	Amruta			ALokh

Sales Home Opportunities Quotes Leads Tasks Files Accounts Contacts Campaigns Dashboards Reports Chatter Groups More

Recently Viewed

17 Items • Updated a few seconds ago

	Name	Account Name	Account Site	Phone	Email	Contact Owner Alias
1	Test2					ALokh
2	Ashish Lokhande	Pooja		1234567890		ALokh
3	Amruta Lokhande	Pooja		1234567890		ALokh
4	con3	Test1		(988) 775-4434		ALokh
5	con2	Test1		(988) 775-4434		ALokh
6	Con 1	Test1		(988) 775-4434		ALokh
7	new Contact	Test				ALokh
8	Amruta Lokhande	Test				ALokh
9	Contact 4	null Contact 4				ALokh
10	Contact 1	Test		(987) 654-3221		ALokh
11	Contact 3			(987) 654-3210		ALokh
12	Contact 2			(987) 654-3210		ALokh
13	Test4			876456787		ALokh

Example 40

Write a trigger on Contact to Prevent the user to create duplicate Contact based on Phone if Phone number is already exist on related account in Salesforce

Trigger: RestrictDuplicatePhoneOnContactTrigger.apxt

```
trigger RestrictDuplicatePhoneOnContactTrigger on Contact
(before insert,before update) {

    if(trigger.isBefore){
        system.debug('Event before trigger');
        if(trigger.isInsert){
            duplicatePhoneOnContactCtrl.
                checkDuplicatePhone(Trigger.new);
        }
        else if(trigger.isUpdate){
            duplicatePhoneOnContactCtrl.
                checkDuplicatePhone(Trigger.new);
        }
    }
    else if(trigger.isAfter){
        system.debug('trigger after event');

    }
}
```



Class: **duplicatePhoneOnContactCtrl.apxc**

```

public class duplicatePhoneOnContactCtrl
{
    public static void checkDuplicatePhone(List<Contact>
                                            conObjItem)
    {

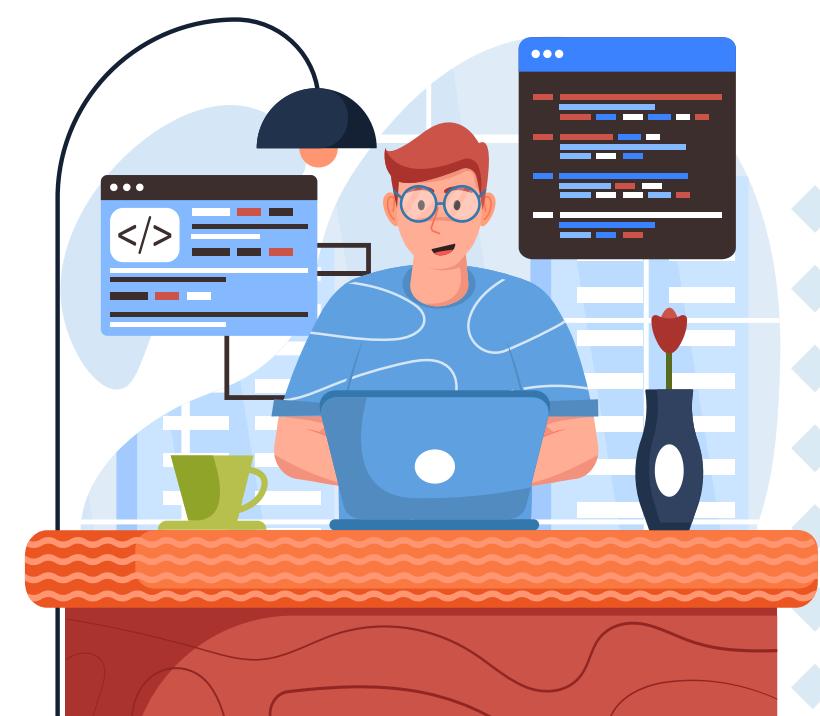
        Map<String, String> conMapItem=new Map<String, String>();
        set<String> accStr=new set<String>();

        for(Contact con:conObjItem){
            if(con.AccountId!=null)
                accStr.add(con.AccountId);
        }

        for(Contact con:[select Id,Name,Phone,AccountId from
                        Contact where AccountId=:accStr])
        {
            conMapItem.put(con.Phone,con.AccountId);
        }

        for(Contact con:conObjItem)
        {
            if( !conMapItem.isEmpty() &&
                conMapItem.containsKey(con.Phone))
            {
                conaddError('Do not allow duplicate Phone
                            number, this phone number is already exist
                            on related account');
            }
        }
    }
}

```





Test Case:

1. Create new Account record and must fill the phone field As shown in below screenshot.

Edit Ankita

Account Owner Amruta Lokhande	Rating Hot
*Account Name Ankita	Phone 9876543210
Parent Account Search Accounts...	Fax
Account Number	Website www.premiumlearnings.com
Account Site	Ticker Symbol
Type Customer - Direct	Ownership Public
Industry Banking	Employees
Annual Revenue	SIC Code
<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>	

Contact Last updated: SIC Code

AI Search Accounts and more..

Sales Home Opportunities Leads Tasks Files Accounts Contacts Campaigns Dashboards Reports Chatter Groups Calendar More

Account Ankita

Type Customer - Direct	Phone (987) 654-3210	Website www.premiumlearnings.com	Account Owner Amruta Lokhande	Account Site	Industry Banking
---------------------------	-------------------------	-------------------------------------	----------------------------------	--------------	---------------------

Related	Details	News
Account Owner Amruta Lokhande	Rating Hot	
Account Name Ankita	Phone (987) 654-3210	
Parent Account	Fax	
Account Number	Website www.premiumlearnings.com	
Account Site	Ticker Symbol	
Type Customer - Direct	Ownership Public	
Industry Banking	Employees	
Annual Revenue	SIC Code	
Contact Last updated		

Activity Chatter

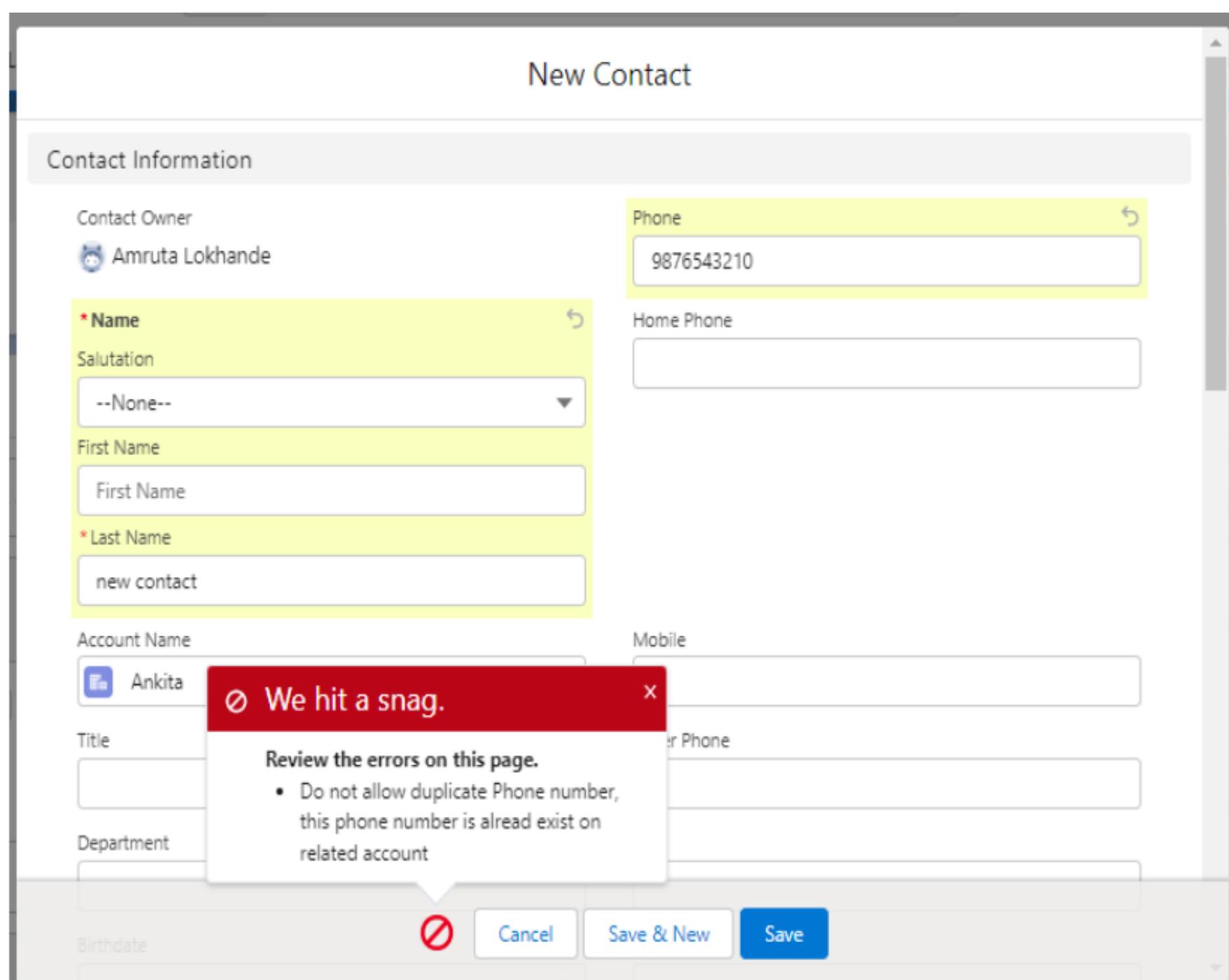
New Task Log a Call New Event Email
Create a task... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

2. try to create child contact record of Account with same phone number, you will get an error as shown in following screenshot.





Example 41

Trigger to Update the Custom Last Modified Date Field in the Parent Object When the Records in the Child Object is Updated Using Apex Trigger in Salesforce

Trigger: UpdateLastModifiedDateOnLr

- Create a custom field on parent Object name as `LastModifiedDate__c` (Date/Time Type).
- Create a lookup relationship field on child Object

```
trigger UpdateLastModifiedDateOnLr on Leave_Request__c
(after insert, after update) {

    Map<Id, Datetime> idLastModifiedDateMap = new
                                Map<Id, Datetime>();
    List<Datetime> childIdList = new List<Datetime>();
    Set<Id> setIdList = new Set<Id>();

    if(trigger.isBefore)
    {
        system.debug('trigger before event');
    }
    else if(trigger.isAfter){
        system.debug('trigger after event');

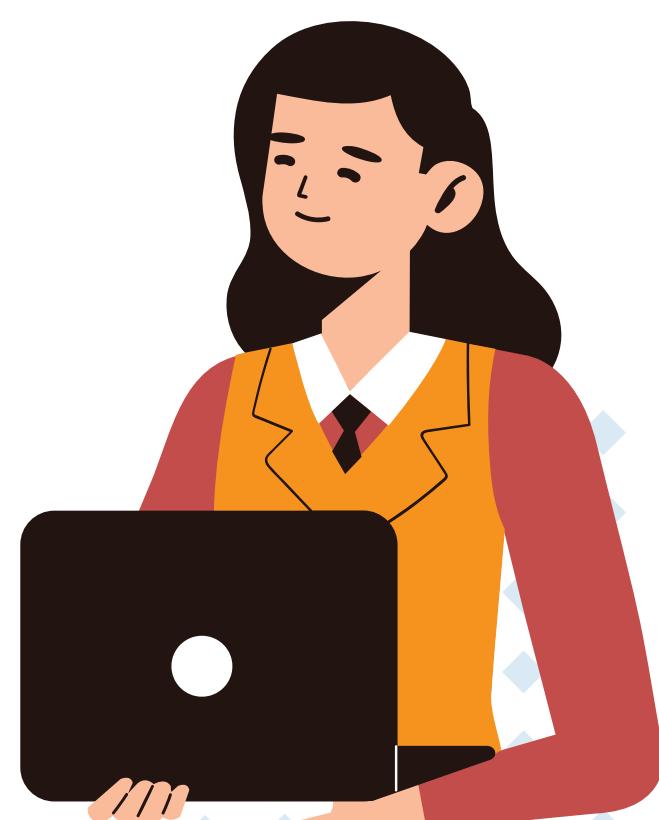
    if(trigger.isUpdate){
        for(Leave_Request__c temp : trigger.new)
        {
            childIdList.add(temp.LastModifiedDate);
        }
    }
}
}
```

```
system.debug('childIdList ' + childIdList);

List<Consultant__c> prntList= [SELECT Id,
                                LastModifiedDate__c From Consultant__c];
List<Consultant__c> lastDate = new List<Consultant__c>();

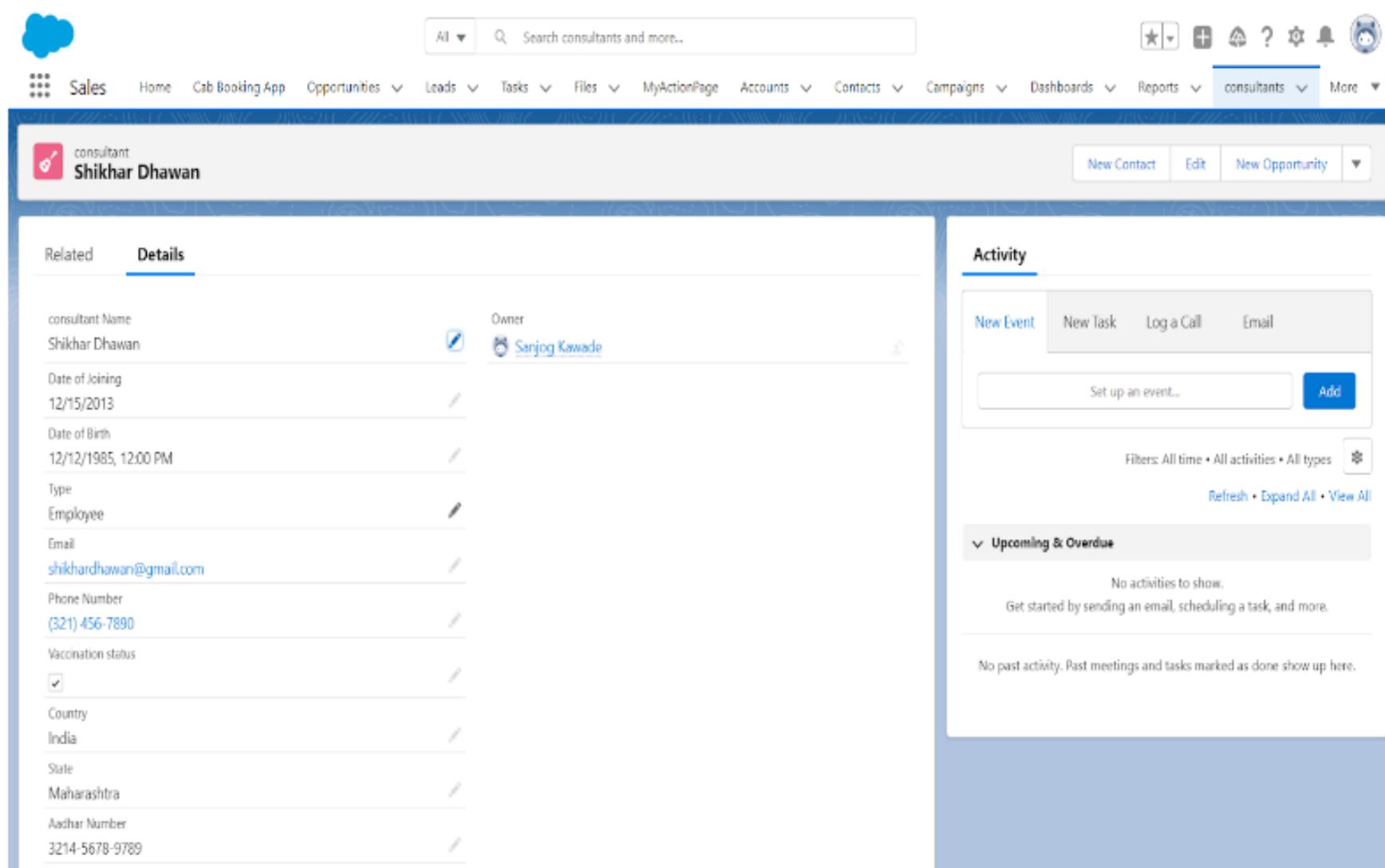
for(Consultant__c p1:prntList){
    for(Leave_Request__c c1: trigger.new){
        system.debug('c1 ' + c1.LastModifiedDate);
        p1.LastModifiedDate__c=c1.LastModifiedDate;
        lastDate.add(p1);
    }
}

update prntList;
system.debug('lastDate1### ' + lastDate);
}
```

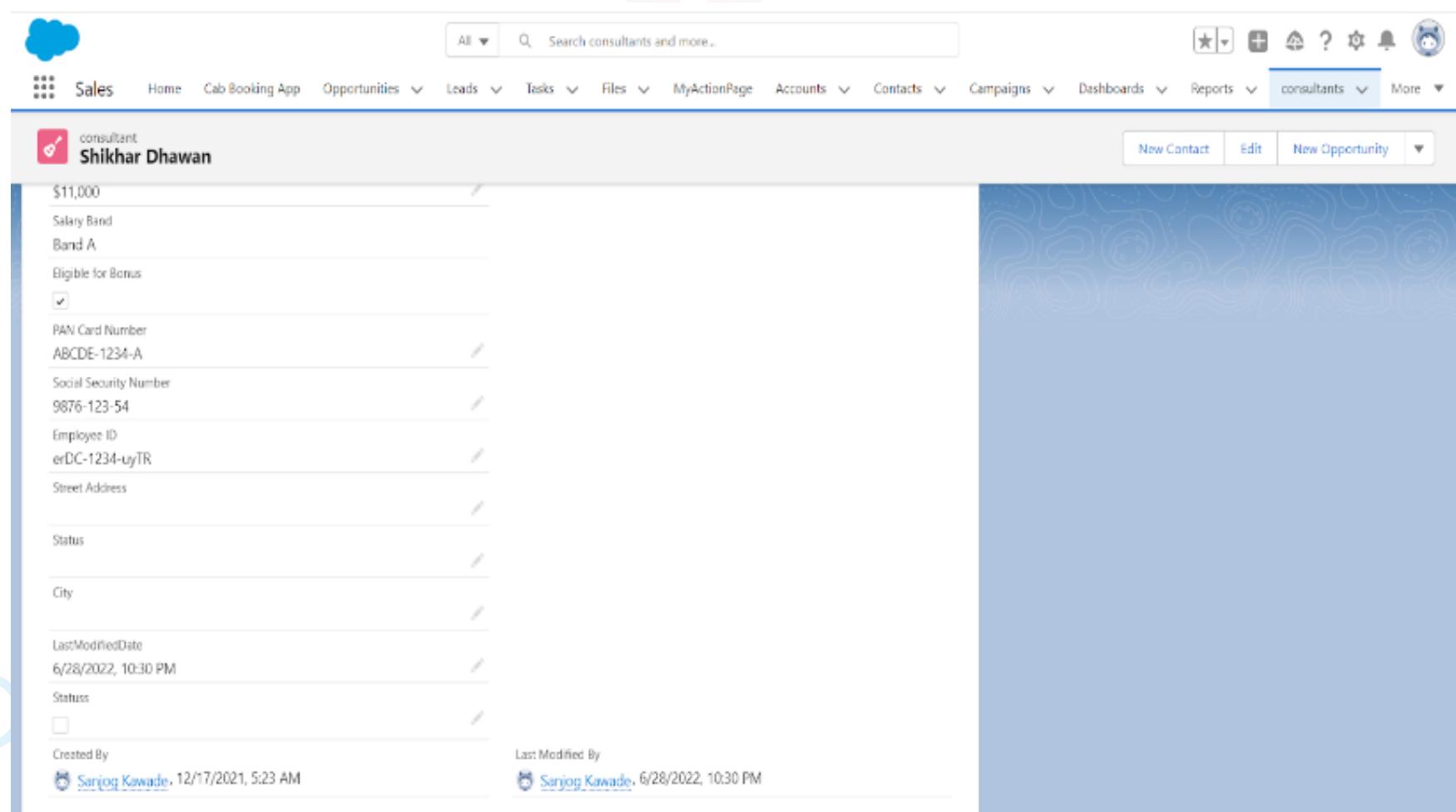


Test Case:

1. take any consultant record which is previously created (Last modified date should be less than today) as shown in below screenshot.



In below screenshot you can see the last modified date of record.





2. Now create new child leave_Request record related to the parent consultant as shown in following screenshot.

New Leave Request

Information

*Leave Request Name	Shikhar Dhawan Leave
*consultant	Shikhar Dhawan
Start Date	7/14/2022
End Date	7/15/2022
Status	<input checked="" type="checkbox"/>

Buttons: Cancel, Save & New, Save

Sales Home Cab Booking App Opportunities Leads Tasks Files MyActionPage Accounts Contacts Campaigns Dashboards Leave Requests More

Leave Request Shikhar Dhawan Leave ✓ Leave Request Shikhar Dhawan Leave was created. X

Details

Leave Request Name	Shikhar Dhawan Leave
consultant	Shikhar Dhawan
Start Date	7/14/2022
End Date	7/15/2022
Status	<input checked="" type="checkbox"/>
Created By	Sanjog Kawade - 7/7/2022, 10:33 PM
Last Modified By	Sanjog Kawade - 7/7/2022, 10:33 PM

Activity

New Event New Task Log a Call Email

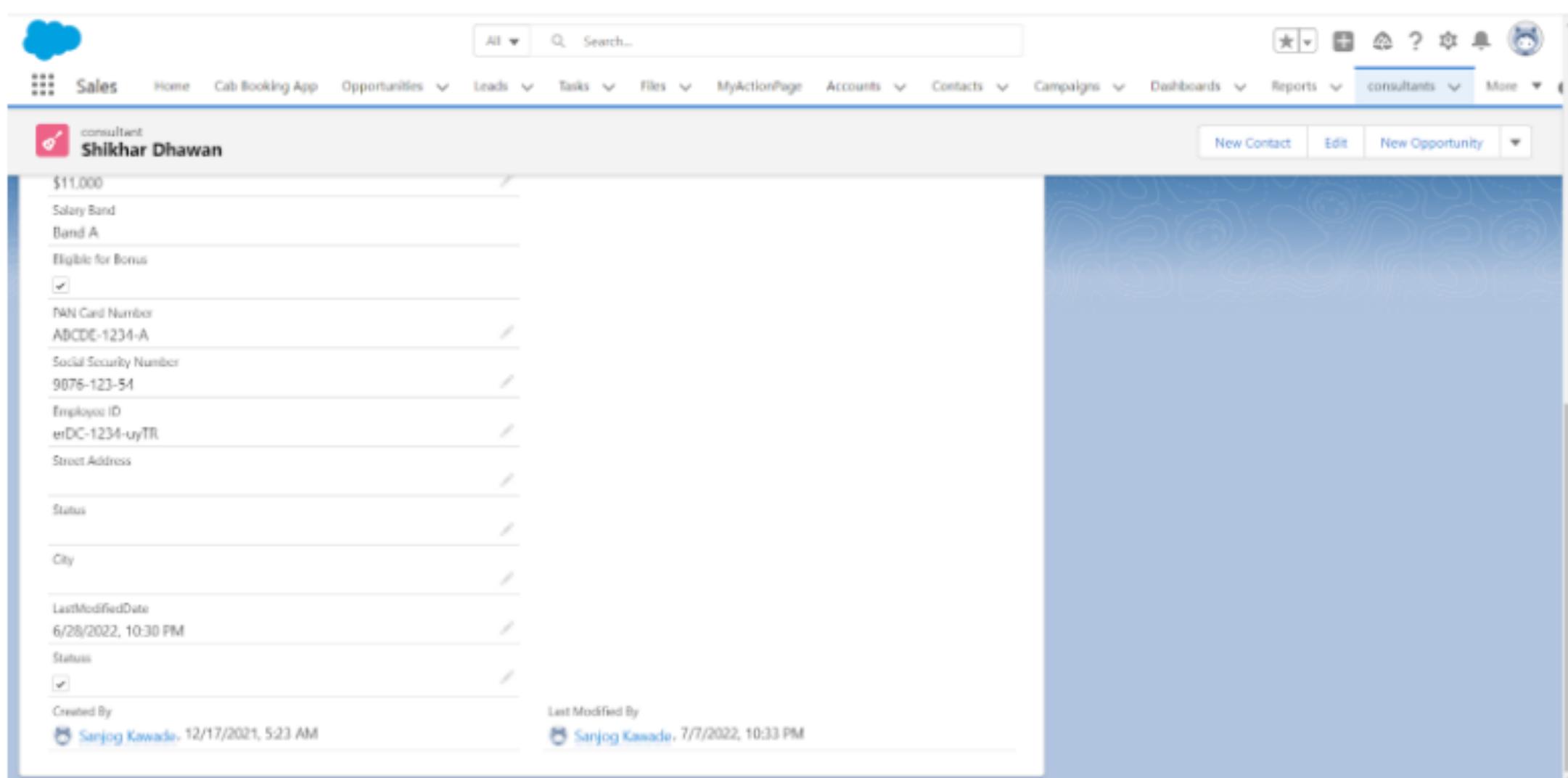
Set up an event... Add

Filters: All time • All activities • All types Refresh • Expand All • View All

No activities to show. Get started by sending an email, scheduling a task, and more.

No past activity. Past meetings and tasks marked as done show up here.

3. Now check the Custom Last Modified Date Field on the Parent Consultant Object will update When the Records in the Child Leave_Request Object is Updated.





Example 42

When we create the Account record, the Account Owner will be automatically added to Sales Repr field. When we update the Account owner of the record, then also the Sales Rep will be automatically updated.

Trigger: UpdateSalesRep.apxt

- Create custom field on Account named Sales_Repr__c(Type Text).

```
trigger UpdateSalesRep on Account (before insert, before update) {
    Set<Id>setAccOwner=new Set<Id>();
    for(Account Acc: trigger.new)
    {
        setAccOwner.add(Acc.OwnerId);
    }

    Map<Id,User> User_map = new Map<Id,User>([select Name
                                                from User where id in:setAccOwner]);
    for(Account Acc: Trigger.new)
    {
        User usr=User_map.get(Acc.OwnerId);
        Acc.Sales_Repr__c=usr.Name;
    }
}
```





Test Case:

1. Create new Account record.

Account Information

Account Owner Amruta Lokhande	Rating Hot	
*Account Name New Account	Phone 9865678244	
Parent Account Search Accounts...	Fax	
Account Number	Website www.premiumlearnings.com	
Account Site	Ticker Symbol	
Type Prospect	Ownership Public	
Industry Consulting	Employees	
Annual Revenue	SIC Code	
Contact Last updated Date Time	Contact Updating count	
<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>		

Annual Revenue	SIC Code	
Contact Last updated Date Time	Contact Updating count	
Premium <input checked="" type="checkbox"/>	Contact	
Support level Premium Plus	Search Contacts...	
No of Open Opportunities 65	Out of Business	
Sales Repr		
Address Information		
Billing Address Billing Street	Shipping Address	
<input type="button" value="Cancel"/> <input type="button" value="Save & New"/> <input type="button" value="Save"/>		

2. check Account Owner will be automatically added to Sales Repr field.

