



# S.K.P ENGINEERING COLLEGE

College in Kilnachipattu, Tamil Nadu 606611.

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BACHELOR OF ENGINEERING

2024 - 2025

### A CRM Application to Handle the Clients and their property Related Requirements

#### TEAM MEMBERS:

- |               |                |
|---------------|----------------|
| A.MANISHA     | - 512222104034 |
| K.MANISHA     | - 512222104035 |
| R.MANOJ KUMAR | - 512222104036 |
| R.MOHAN RAJ   | - 512222104037 |

# S.K.P ENGINEERING COLLEGE

College in Kilnachipattu, Tamil Nadu 606611.



## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certified that this is a bonafide record of work done by

Name NM ID : MANISHA.A

University : 4CD732C265302BD90F54D4CDD6E75340

Reg.No : 512222104034

Semester : 5

Branch : CSE

Year : 2024-2025

Staff-in-Charge Head of the Department

Submitted for the  
Practical Examination held on

Internal Examiner

External Examiner

# A CRM Application to Handle the Clients and their property Related Requirements

## PROJECT OVERVIEW:

Dreams World Properties integrates Salesforce to streamline customer interactions. Website engagement triggers automated record creation in Salesforce, capturing customer details and preferences. Salesforce categorizes users as approved or non-approved, offering tailored property selections to approved users. This enhances user experience and efficiency, providing personalized recommendations and broader listings. Seamless integration optimizes operations, improving customer engagement and facilitating growth in the real estate market.

## Objectives : -

### 1) Website Integration Requirements :

Implement a form on the website for users to express interest in property listings.  
Ensure the form captures essential details such as name, contact information, preferred property type, location, budget, etc. Set up validation rules to ensure data accuracy and completeness.  
Integrate the form submission process with Salesforce.

### 2) Salesforce Configuration Requirements :

Set up Salesforce objects and fields to store customer data. This includes fields for name, contact information, preferences, approval status, etc. Define workflows or processes to automate the creation of records when a user submits the form on the website. Configure Salesforce security settings to control access to customer records based on approval status.

### 3) Approval Process Requirements :

Define criteria for categorizing users as approved or non-approved based on specific parameters such as budget, property preferences, etc. Implement an approval process in Salesforce to review and approve users. Set up email notifications or alerts to notify relevant stakeholders when a user is approved or rejected. Ensure that approved users are granted access to curated property listings tailored to their preferences.

### 4) User Experience Requirements:

Design user interfaces in Salesforce for managing customer records, approval processes, and property listings. Ensure a seamless user experience for both customers and internal users interacting with Salesforce.

## 5) Integration Testing and Quality Assurance Requirements:

Conduct thorough testing of the integration between the website and Salesforce to ensure data is accurately captured and transferred. Perform end-to-end testing of the approval process to verify that users are categorized correctly and granted appropriate access.

Identify and resolve any issues or bugs encountered during testing.

## 6) Scalability and Performance Requirements:

Ensure that the integration and Salesforce configuration are scalable to accommodate future growth in customer volume and data. Optimize system performance to ensure responsiveness and reliability, especially during peak usage periods.

## 7) Documentation and Maintenance Requirements:

Document the integration architecture, data flows, and configuration details for future reference. Establish procedures for ongoing maintenance and updates to the integration and Salesforce configuration. Provide ongoing support and training for users to address any issues or questions that may arise.

## **Salesforce key feature and concept utilized:**

1. Real Time Salesforce Project

2. Object & Fields

3. Integration Through Jotform

4. Roles

5. Application Management

6. Profiles

7. UserManagement

8. Approval Process

9. Flows

10. LWC Components

## Detailed Steps to Solution Design:

- To create a CRM Application to handle the clients and their property related requirement we need the following steps

Milestone 1:- Create a jotform and integrate it with the org to create a record of customers automatically.

USE CASE :- Client wants a form for the customers to get the details directly into the salesforce so that the admins can create a user in the org.

- 1) Open your browser and search for jotform and login.

Google

All Images Videos News Shopping More Tools

About 77,70,000 results (0.29 seconds)

 Jotform  
<https://www.jotform.com> ::

**Jotform: Free Online Form Builder & Form Creator**

Create forms and surveys for free with Jotform's drag-and-drop form builder. Start collecting registrations, applications, orders, and payments today.

Results from jotform.com

**Login**   
Jotform's form builder helps you create & publish online forms ...

**My Forms**  
Access and manage your forms and submissions on Jotform's ...

**Pricing**  
Whether you need a robust online form solution, or just the basics ...

**Signup**  
If you need online forms for generating leads, distributing ...

2) After log in click on create form and click on start from scratch

 Jotform My Forms

My Forms Templates Integrations Products Support Enterprise Pricing Login Sign Up For Free

**MY FORMS**

-  All Forms
-  Create a new folder

**MY TEAMS**

-  Create Team

**SHARED WITH ME**

**ASSIGNED FORMS**

My Drafts



YOU DON'T HAVE ANY FORMS YET  
Your forms will appear here.

3) Now create a form to get the customer details like Name, Phone, Email, Address and type of property the customer is interested in.

Dreams World  
Last edited yesterday. 0

BUILD SETTINGS PUBLISH



### Dreams World

Name \*

First Name  Last Name

Email

example@example.com

Phone Number

(000) 000-0000

Please enter a valid phone number.

Which type of Property are you looking for?

RESIDENTIAL  
 COMMERCIAL  
 RENTAL

Budget Amount \*

e.g. 23

Address

Street Address  
 Street Address Line 2  
 City  State / Province  
 Postal / Zip Code

**Submit**

+ ADD NEW PAGE HERE

If you want to remove Jotform Branding, [please upgrade your account](#)

 [Now create your own Jotforms - It's Free](#) [Create your own Jotforms](#)

4) Once the form is created, publish it by clicking on publish.

<https://www.jotform.com/form/240031134484041>

## Milestone 2 :- Create Objects from Spreadsheet.

### ● Create Customer object

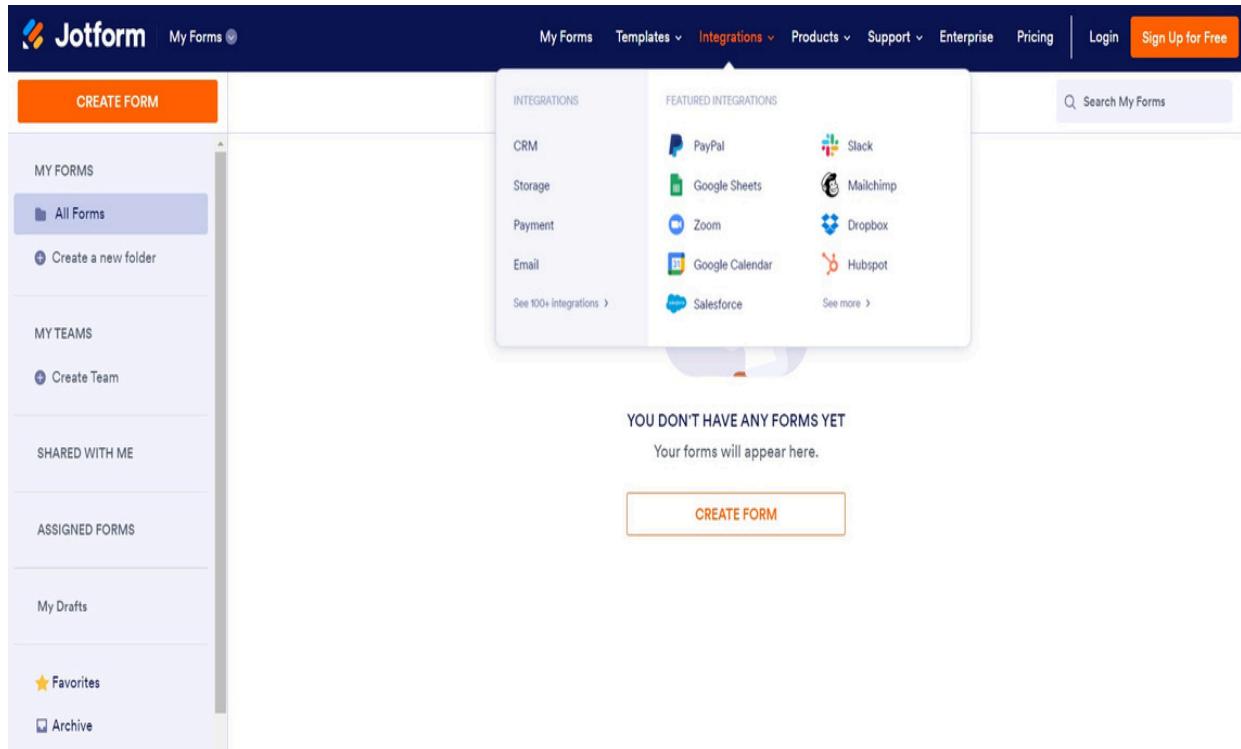
- 1) Go to your object manager and click on create object from spreadsheet
- 2) Click on the link to get the spreadsheet,
- 3) [customer](#)
- 4) After downloading, upload the file, map the fields and upload to create an object.

### ● Create Property object

- 1) Follow the same from the customer object to create the Property Object
- 2) [Property](#)

## Milestone 3 : - Integrate Jotform with Salesforce Platform ]

- 1) On the Jotform Platform, Click on Integration and choose Salesforce.



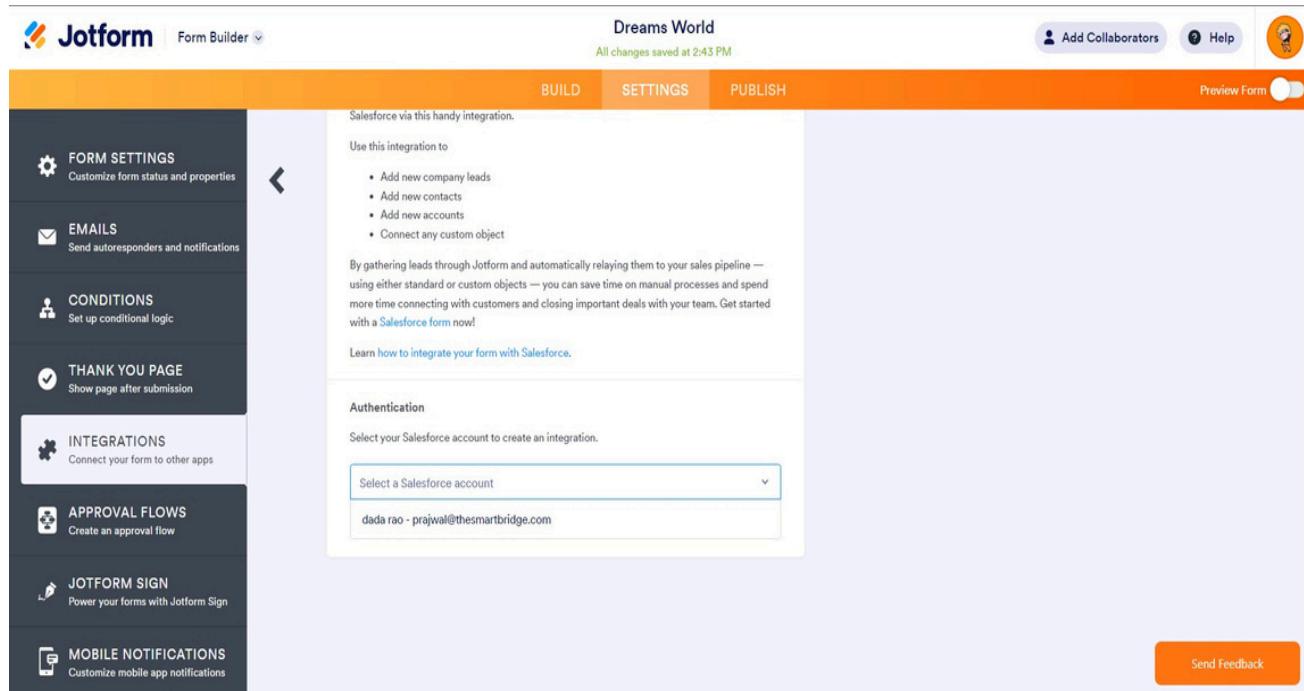
The Jotform homepage features a navigation bar with links for My Forms, Templates, Integrations, Products, Support, Enterprise, Pricing, Login, and Sign Up for Free. A prominent orange "CREATE FORM" button is located on the left sidebar. The main content area displays a section titled "INTEGRATIONS" with categories like CRM, Storage, Payment, and Email, each with a list of featured integrations such as PayPal, Slack, Google Sheets, Mailchimp, Zoom, Dropbox, Google Calendar, Hubspot, and Salesforce. Below this is a message stating "YOU DON'T HAVE ANY FORMS YET" with the subtext "Your forms will appear here." and another "CREATE FORM" button.

2) Click on User Integration and choose “Add to Form”.



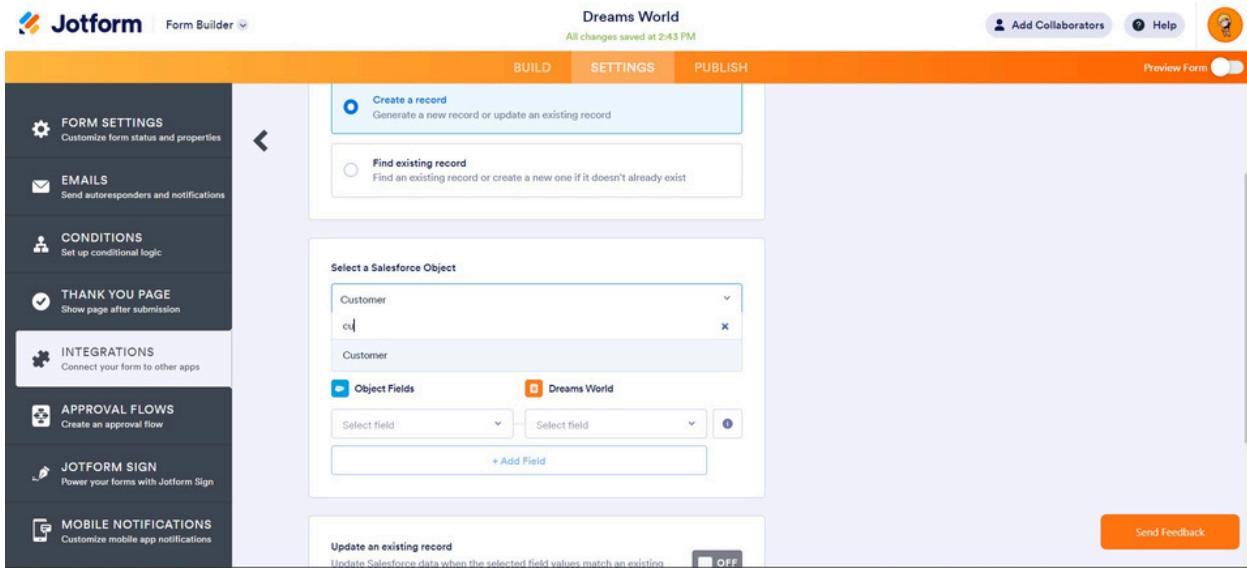
The screenshot shows the Salesforce integration page for Jotform. It features a "Watch Video" button and a "Use Integration" dropdown menu with "Go to AppExchange" and "Add to Form" options, the latter being highlighted. The main visual illustrates a "Sales Contact Form" on the left and a "Contact Details" page on the right, showing how data is transferred between them. The Jotform and Salesforce logos are at the bottom, along with the text "Send new leads, contacts, or accounts to your sales CRM". A descriptive paragraph at the bottom explains that Salesforce is a powerful CRM tool used to manage accounts, improve client relations, and close deals, with a note about automatic sync with Jotform submissions.

3) Select the Org with which you want to Integrate your jotform with.



The screenshot shows the Jotform Form Builder interface. The left sidebar has a dark theme with various options like FORM SETTINGS, EMAILS, CONDITIONS, THANK YOU PAGE, APPROVAL FLOWS, JOTFORM SIGN, and MOBILE NOTIFICATIONS. The main area is titled 'Dreams World' and shows the 'INTEGRATIONS' section for Salesforce. It includes a heading 'Salesforce via this handy integration.', a list of actions ('Add new company leads', 'Add new contacts', 'Add new accounts', 'Connect any custom object'), a descriptive text about connecting leads to sales pipeline, and a link to 'Learn how to integrate your form with Salesforce.'. Below this is an 'Authentication' section with a dropdown menu 'Select a Salesforce account' containing 'dada rao - prajwal@thesmartbridge.com'. At the bottom right is an orange 'Send Feedback' button.

4) Select an Action - Create a record.  
Select a Salesforce Object : - Customer



The screenshot shows the Jotform Form Builder interface with the 'INTEGRATIONS' section for Salesforce selected. The main area displays the 'Create a record' action, which generates a new record or updates an existing one. Below it is the 'Find existing record' action. A large modal window titled 'Select a Salesforce Object' is open, showing a dropdown menu with 'Customer' selected. Under 'Object Fields', there are two dropdowns labeled 'Select field' and 'Dreams World', with a 'Select field' button and a '+ Add Field' button. At the bottom of the modal is an 'Update an existing record' section with a note about matching field values and a toggle switch set to 'OFF'. The bottom right of the modal has an orange 'Send Feedback' button.

5) Map Each and every field on the Object with the fields on the form and “Save Action”.

Create a record  
Send data from form fields to matched Salesforce fields

Object Fields	Dreams World
Customer	Name - First Name
City	Address - City
Budget Amount	Budget Amount
Property Type	Which type of Property ar...
Phone Number	Phone Number
Street Address	Address - Street Address
Email	Email
Customer Name	Name - Last Name
State	Address - State
Street Address line 2	Address - Street Address 2

+ Add Field

6) Then “Save the Integration” and “Finish”.

SALESFORCE  
Send new leads, contacts, or accounts to your sales CRM

All Actions  + Add New Action

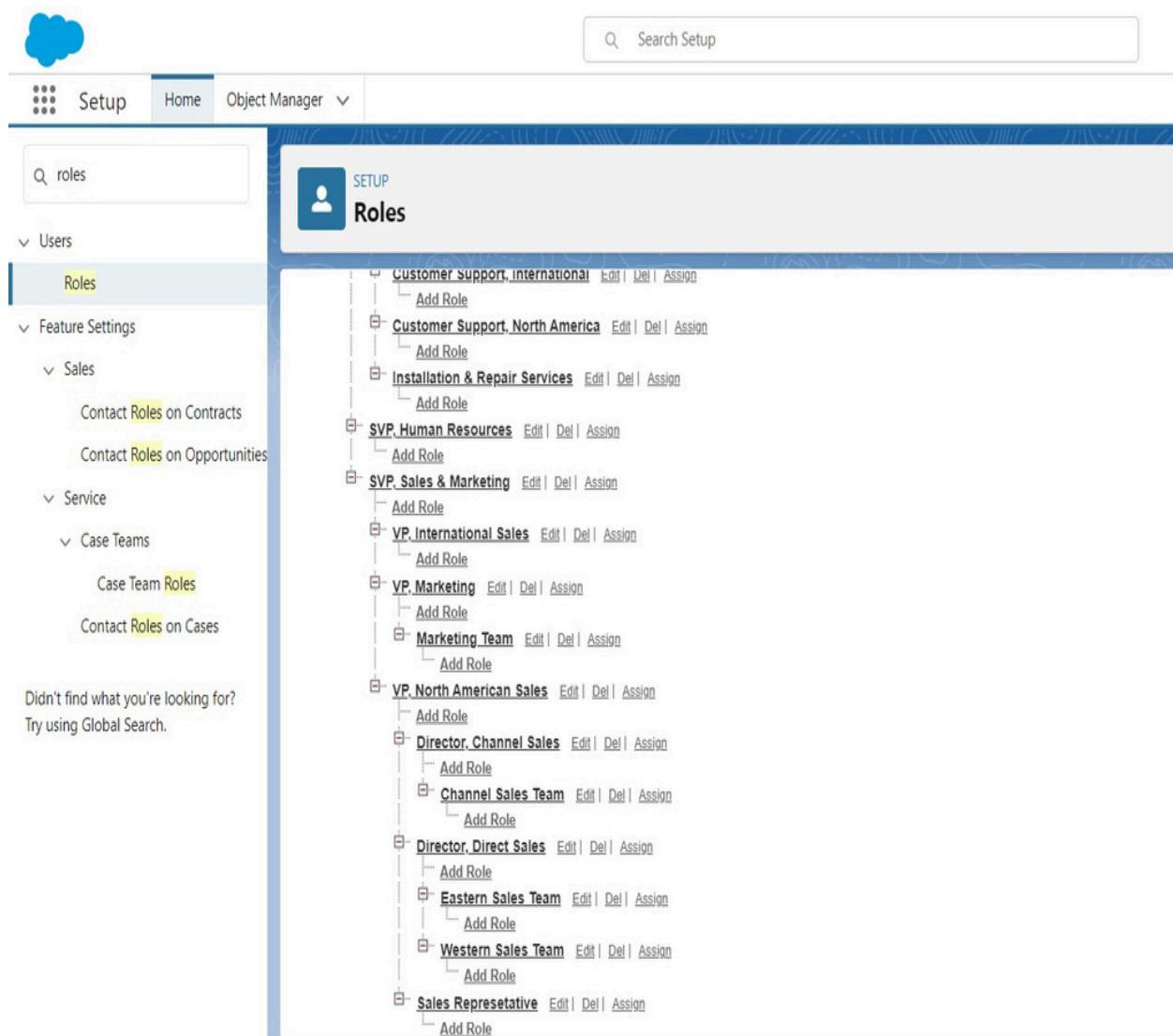
1 Create a record  
Customer

CANCEL 

## Milestone 4 : Create Roles

- SalesExecutiveRole

- 1) Go to Setup and Click on Roles, then click on Expand all and Add a Role just below the Sales Representative

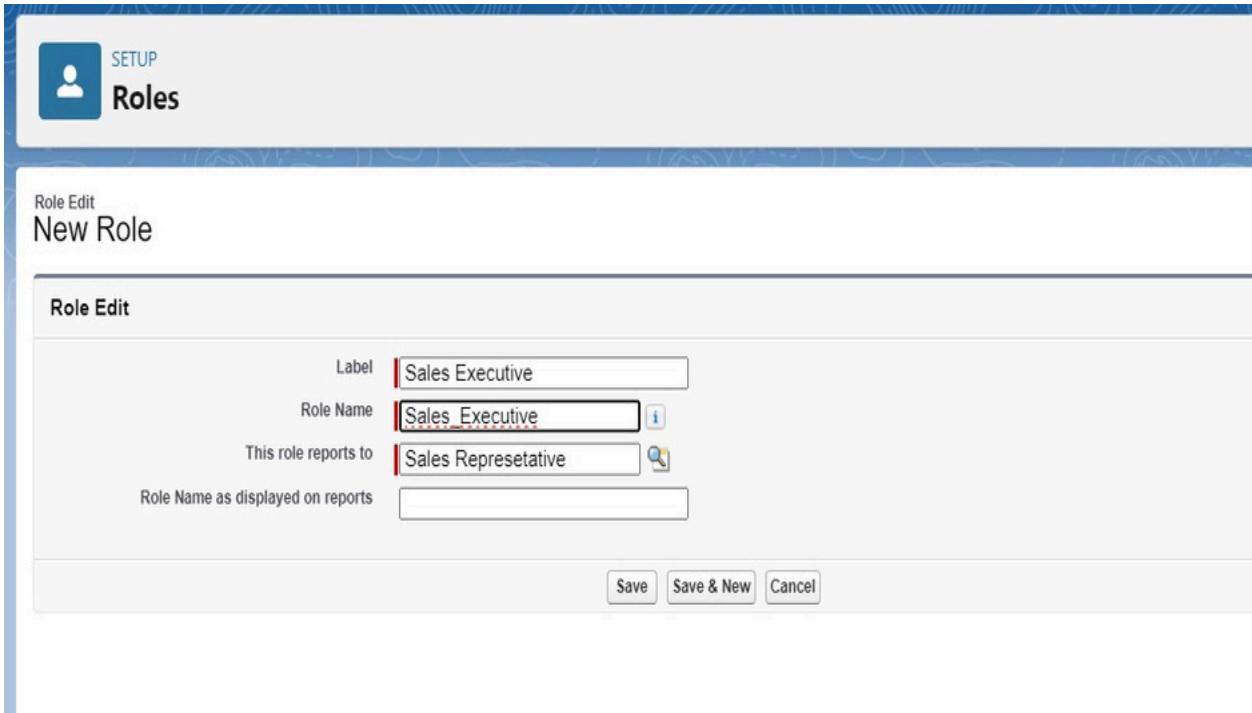


The screenshot shows the Salesforce Setup interface with the 'Roles' page selected. The left sidebar shows navigation categories like Users, Feature Settings, Sales, Service, and Case Teams, with 'Roles' currently highlighted. The main content area displays a hierarchical list of roles under the 'Customer Support' category. Each role entry includes 'Edit', 'Del', and 'Assign' buttons. The roles listed are:

- Customer Support, International
  - Add Role
- Customer Support, North America
  - Add Role
- Installation & Repair Services
  - Add Role
- SVP\_Human Resources
  - Add Role
- SVP\_Sales & Marketing
  - Add Role
- VP\_International Sales
  - Add Role
- VP\_Marketing
  - Add Role
- Marketing Team
  - Add Role
- VP\_North American Sales
  - Add Role
- Director\_Channel Sales
  - Add Role
- Channel Sales Team
  - Add Role
- Director\_Direct Sales
  - Add Role
- Eastern Sales Team
  - Add Role
- Western Sales Team
  - Add Role
- Sales Represatative
  - Add Role

\* It will use the “System Administrator Profile”.

- 2) Label -SalesExecutive  
Reports to - Sales Representative



The screenshot shows the Salesforce 'Roles' page under the 'SETUP' tab. A new role is being created with the following details:

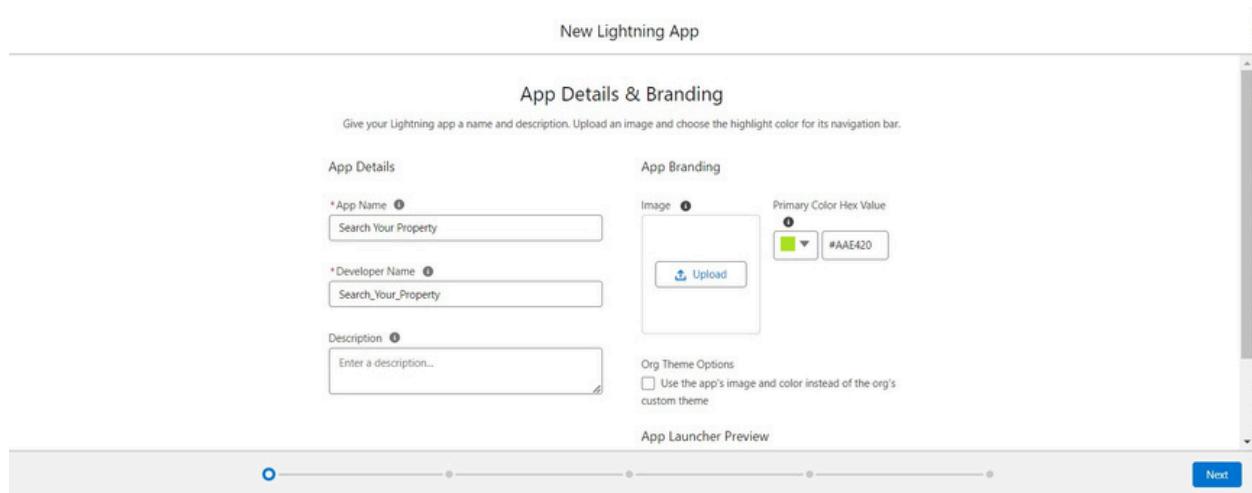
Role Edit	
Label	Sales Executive
Role Name	Sales_Executive
This role reports to	Sales Represetative
Role Name as displayed on reports	(empty)

At the bottom of the form are three buttons: 'Save', 'Save & New', and 'Cancel'.

- Similarly Create a Role Name "Sales Manager" below Sales Executive which reports to Sales Executive, Also Add a Role below Sales Manager labeled as "Customer" which reports to Sales Manager.

## Milestone 5 : - Create a Property Details App

- 1) From Setup → Go to App Manager and click on New Lightning App and Name it as "Property Details" and add "Customer" and "Property" Object.

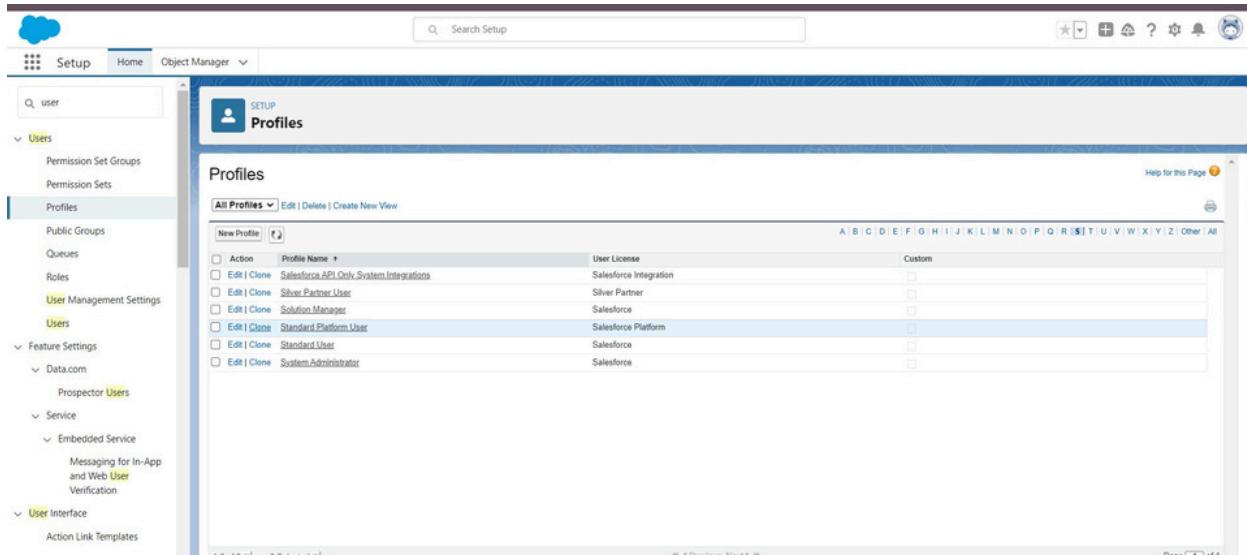


- 2) Click Next → Next → Save and Add "System Admin "Profile.

## Milestone 6 : - Create Profiles

- Customer :-

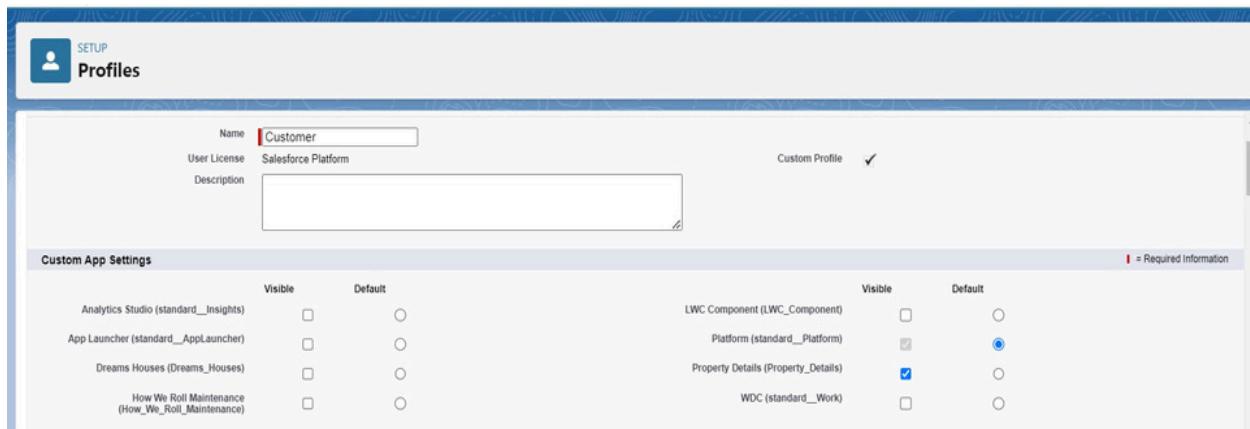
- 1) From Setup → Go to Profiles and Clone Salesforce Platform User and Name it "Customer" ..



The screenshot shows the Salesforce Setup interface with the 'Profiles' page open. The left sidebar is expanded to show 'Users' and 'Feature Settings'. The main area displays a list of profiles. The 'Standard Platform User' profile is selected. The table columns are: Action, Profile Name, User License, and Custom.

Action	Profile Name	User License	Custom
<input type="checkbox"/> Edit   Clone	Salesforce API Only System Integrations	Salesforce Integration	<input type="checkbox"/>
<input type="checkbox"/> Edit   Clone	Silver Partner User	Silver Partner	<input type="checkbox"/>
<input type="checkbox"/> Edit   Clone	Solution Manager	Salesforce	<input type="checkbox"/>
<input type="checkbox"/> Edit   Clone	Standard Platform User	Salesforce Platform	<input checked="" type="checkbox"/>
<input type="checkbox"/> Edit   Clone	Standard User	Salesforce	<input type="checkbox"/>
<input type="checkbox"/> Edit   Clone	System Administrator	Salesforce	<input type="checkbox"/>

2) Uncheck all the Custom Objects and Check only Property Details From Custom App Settings.



The screenshot shows the 'Customer' profile edit screen. Under 'Custom App Settings', the 'Property Details (Property\_Details)' object is checked under 'Visible' and selected as 'Default'.

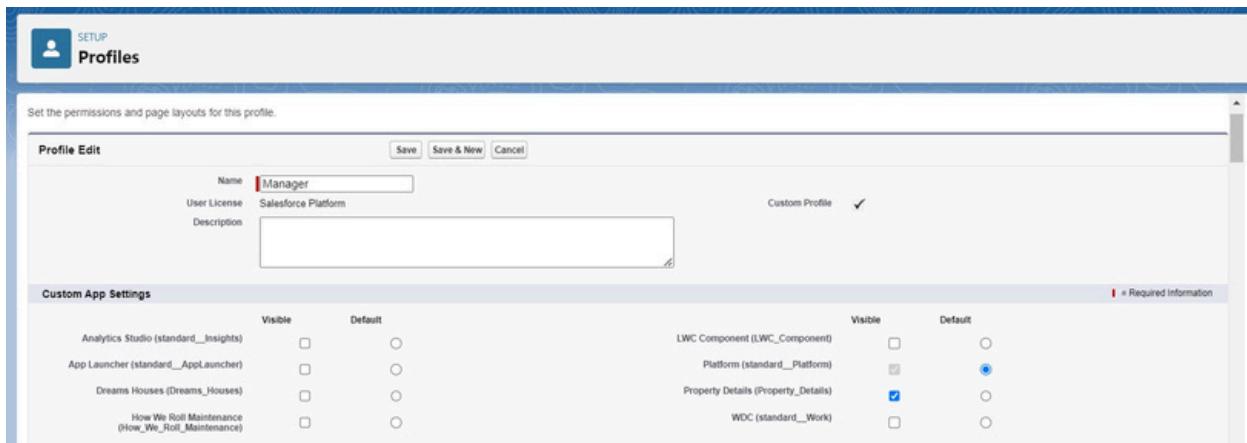
Object	Visible	Default
Analytics Studio (standard__Insights)	<input type="checkbox"/>	<input type="radio"/>
App Launcher (standard__AppLauncher)	<input type="checkbox"/>	<input type="radio"/>
Dreams Houses (Dreams__Houses)	<input type="checkbox"/>	<input type="radio"/>
How We Roll Maintenance (How_We_Roll_Maintenance)	<input type="checkbox"/>	<input type="radio"/>
LWC Component (LWC_Component)	<input type="checkbox"/>	<input type="radio"/>
Platform (standard__Platform)	<input type="checkbox"/>	<input checked="" type="radio"/>
Property Details (Property_Details)	<input checked="" type="checkbox"/>	<input type="radio"/>
WDC (standard__Work)	<input type="checkbox"/>	<input type="radio"/>

3) Also Remove all the Standard Object Permissions.

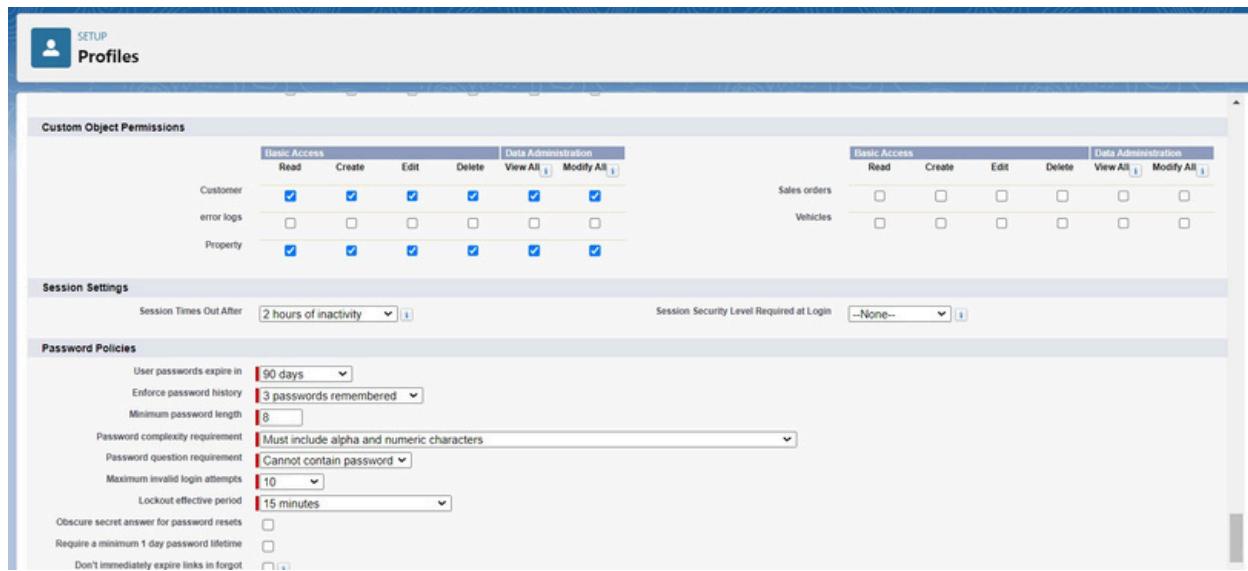


● Manager:-

- 1) From Setup→ Go to Profiles and Clone Salesforce Platform User and Name it “Manager”..
- 2) Uncheck all the Custom Objects and Check only Property Details From Custom App Settings.



- 3) Also Remove all the Standard Object Permissions.
- 4) Uncheck all the Custom Object Permissions and check only“modify all ”from “Property” and “Customer”



**Custom Object Permissions**

	Customer	Sales orders										
	Read	Create	Edit	Delete	View All	Modify All	Read	Create	Edit	Delete	View All	Modify All
Customer	<input checked="" type="checkbox"/>	<input type="checkbox"/>										
error logs	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
Property	<input checked="" type="checkbox"/>	<input type="checkbox"/>										

**Session Settings**

Session Times Out After: 2 hours of inactivity

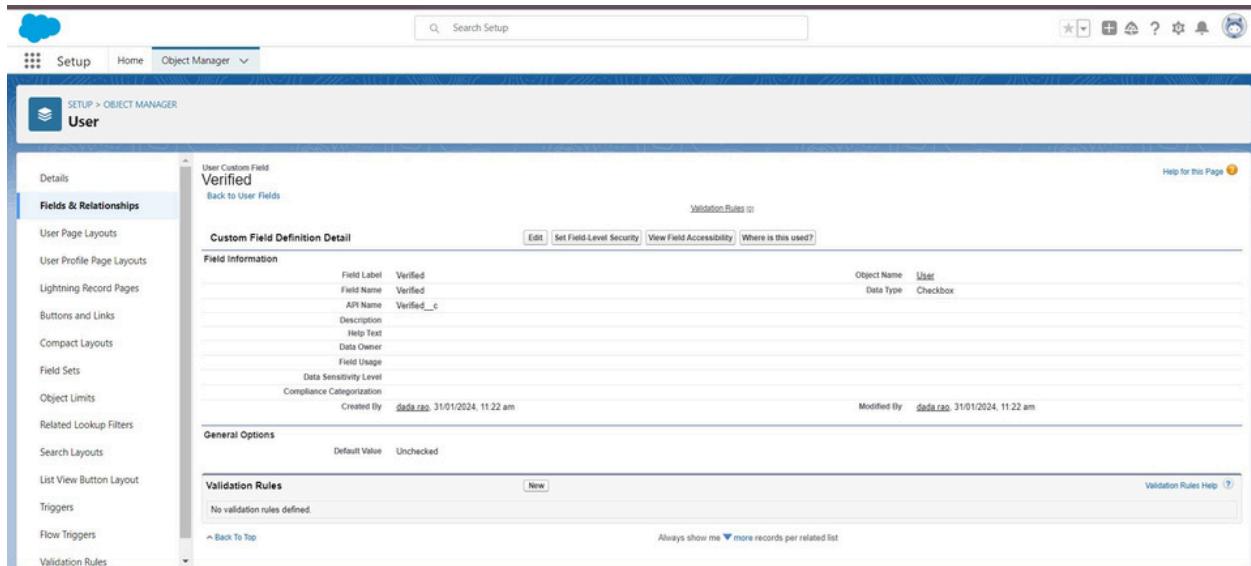
Session Security Level Required at Login: --None--

**Password Policies**

- User passwords expire in: 90 days
- Enforce password history: 3 passwords remembered
- Minimum password length: 6
- Password complexity requirement: Must include alpha and numeric characters
- Password question requirement: Cannot contain password
- Maximum invalid login attempts: 10
- Lockout effective period: 15 minutes
- Obscure secret answer for password resets:
- Require a minimum 1 day password lifetime:
- Don't immediately expire links in forgot:

## Milestone 7 : - Create a CheckBox field on user

- 1) Setup → Object Manager → Search for User → Fields and Relationships
- 2) Create new Field Named as “Verified” as Datatype “CheckBox”



**User Custom Field**  
Verified

**Custom Field Definition Detail**

Field Information	Validation Rules
Field Label: Verified Field Name: Verified API Name: Verified_c Description: <input type="text"/> Help Text: <input type="text"/> Data Owner: <input type="text"/> Field Usage: <input type="text"/> Data Sensitivity Level: <input type="text"/> Compliance Categorization: <input type="text"/> Created By: data_rao, 31/01/2024, 11:22 am	Object Name: User Data Type: Checkbox
General Options	Default Value: Unchecked
Validation Rules	No validation rules defined.

**Fields & Relationships**

- Details
- User Page Layouts
- User Profile Page Layouts
- Lightning Record Pages
- Buttons and Links
- Compact Layouts
- Field Sets
- Object Limits
- Related Lookup Filters
- Search Layouts
- List View Button Layout
- Triggers
- Flow Triggers
- Validation Rules

## Milestone 8 : - Create Users

Create three different users with three different Roles and profiles as we have mentioned above.

User 1 : -

- 1) GotoSetup→Administration→Users→NewUser
- 2) LastName -Executive
- 3) Role-SalesExecutive
- 4) License- Salesforce
- 5) Profile-SystemAdministrator
- 6) Save

User 2 : -

- 1) GotoSetup→Administration→Users→NewUser
- 2) LastName -Manager
- 3) Role-SalesManager
- 4) License- SalesforcePlatform
- 5) Profile-Manager
- 6) Save

User 3 : -

- 1) GotoSetup→Administration→Users→NewUser

- 2) LastName -Customer
- 3) Role-Customer
- 4) License- SalesforcePlatform
- 5) Profile-Customer
- 6) Make Sure the verified checkbox is “Unchecked”
- 7) Save

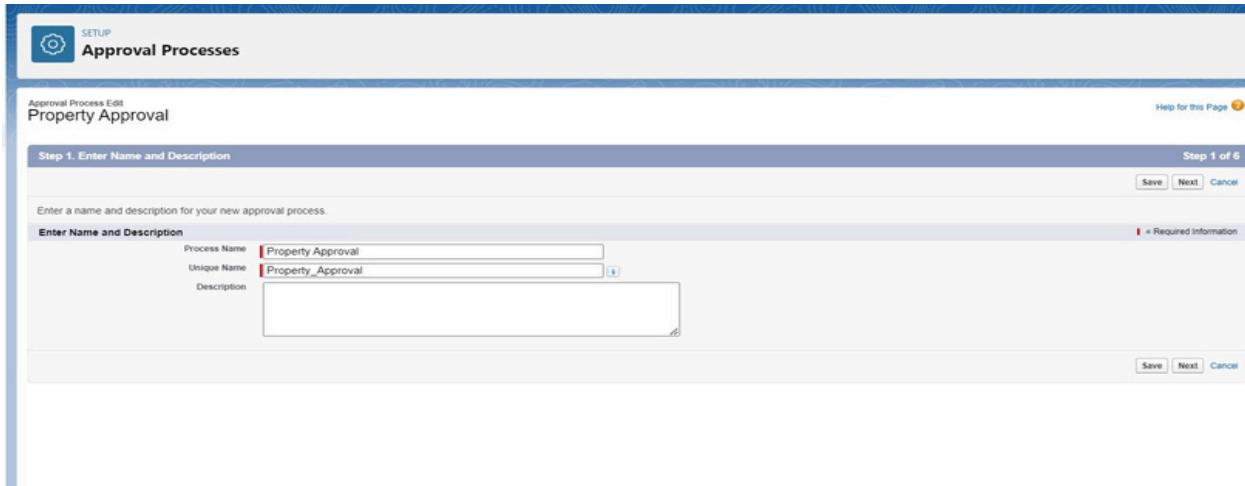
User 4 :-

- 1) Goto Setup → Administration → Users → New User
- 2) LastName -Customer2
- 3) Role-Customer
- 4) License- SalesforcePlatform
- 5) Profile-Customer
- 6) Make Sure the verified checkbox is “checked”
- 7) Save

## Milestone 9 :- Create an Approval Process for Property Object

- 1) From Setup → Process Automation → Approval Process

## 2) Process Name-Property Approval



**Step 1. Enter Name and Description**

Enter a name and description for your new approval process.

**Enter Name and Description**

Process Name	Property Approval
Unique Name	Property_Approval
Description	[Empty]

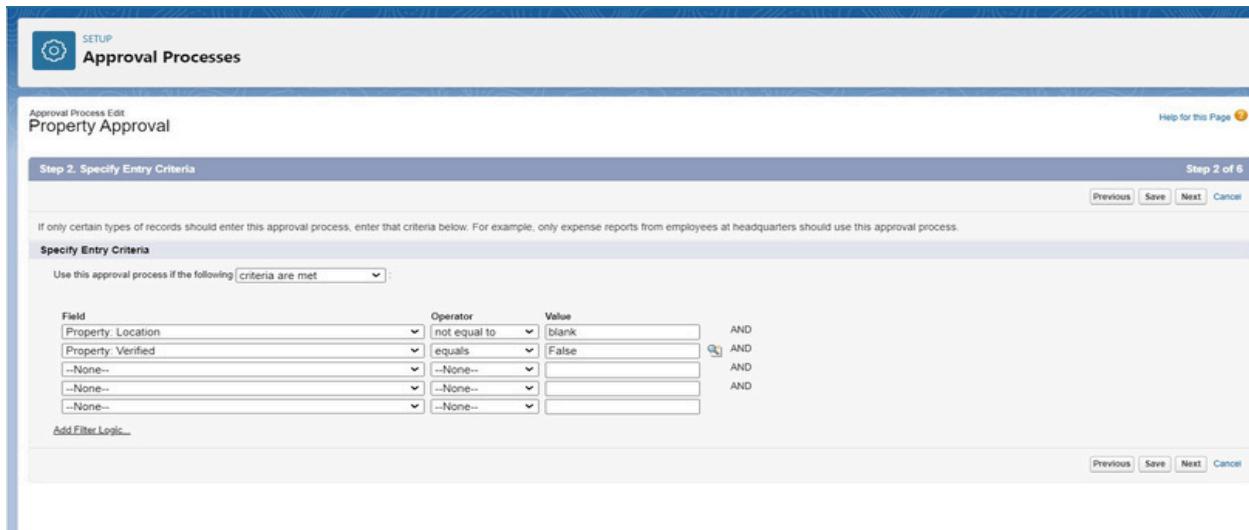
**Step 1 of 6**

Save Next Cancel

= Required Information

## 3) Give 2 criteria→

- a) Location is not equal to blank, b)  
Verified Equals false.



**Step 2. Specify Entry Criteria**

If only certain types of records should enter this approval process, enter that criteria below. For example, only expense reports from employees at headquarters should use this approval process.

**Specify Entry Criteria**

Use this approval process if the following criteria are met :

Field	Operator	Value
Property_Location	not equal to	blank
Property_Verified	equals	False
--None--	--None--	--None--
--None--	--None--	--None--
--None--	--None--	--None--

AND  
AND  
AND

**Step 2 of 6**

Previous Save Next Cancel

4) Click next and “ Next Automated Approver Determined By” → Select Manager

5) From Record Editability Properties → Click on Administrators OR the currently assigned approver can edit records during the approval process.

**SETUP Approval Processes**

Approval Process Edit  
**Property Approval**

Step 3. Specify Approver Field and Record Editability Properties Step 3 of 6

When you define approval steps, you can assign approval requests to different users. One of your options is to use a user field to automatically route these requests. If you want to use this option for any of your approval steps, select a field from the picklist below. Also, when a record is in the approval process, it will always be locked-- only an administrator will be able to edit it. However, you may choose to also allow the currently assigned approver to edit the record.

**Select Field Used for Automated Approval Routing**

Next Automated Approver Determined By: Manager

Use Approver Field of Property Owner:

**Record Editability Properties**

Administrators ONLY can edit records during the approval process.  
 Administrators OR the currently assigned approver can edit records during the approval process.

Previous Save Next Cancel

6) From Step 5. Select Fields to Display on Approval Page Layout select Property, Owner, Location, Type.

**SETUP Approval Processes**

Property Approval

Step 5. Select Fields to Display on Approval Page Layout Step 5 of 6

The approval page is where an approver will actually approve or reject a request. Using the options below, choose the fields to display on this page.

Available Fields	Selected Fields
Created By Last Modified By Property link Verified	Property Owner Location Property Name Type
Add <input type="button" value="Up"/> <input type="button" value="Down"/> Remove	Up <input type="button" value="Down"/> Down

[Click here to view an example](#)

Previous Save Next Cancel

7) Click Next and Select the initial Submitters →

- a ) Owner → Property Owner
- b) Roles→SalesManager

8) Save.

9) Add an approval step name “Executive Approval ”

Approval Step Edit  
VP Approval Help for this Page 

Step 1. Enter Name and Description Step 1 of 3

Enter a name, description, and step number for your new approval step.

**Enter Name and Description** ! = Required Information

Approval Process Name	Property Approval
Name	<input type="text" value="VP Approval"/>
Unique Name	<input type="text" value="VP_Approval"/>
Description	<input type="text"/>

Save Next Cancel

10) specify the Criteria → All record should enter

Approval Step Edit  
VP Approval Help for this Page 

Step 2. Specify Step Criteria Step 2 of 3

Specify whether a record must meet certain criteria before entering this approval step. If these criteria are not met, the approval process can skip to the next step, if one exists. [Learn more](#)

**Specify Step Criteria**

All records should enter this step.  
 Enter this step if the following criteria are met  , else  :

Previous Save Next Cancel

11) click next and select the Approver as “ Sales Executive “ and “Save”

Approval Step Edit  
**VP Approval**

Step 3. Select Assigned Approver Step 3 of 3

Specify the user who should approve records that enter this step. Optionally, choose whether the approver's delegate is also allowed to approve these requests.

**Select Approver**

Let the submitter choose the approver manually.  
 Automatically assign using the user field selected earlier. (Manager)  
 Automatically assign to queue.   
 Automatically assign to approver(s).

User

Add Row Remove Row

When multiple approvers are selected:  
 Approve or reject based on the FIRST response.  
 Require UNANIMOUS approval from all selected approvers.

The approver's delegate may also approve this request.

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

## 12) Add One field Update as “Verified Property”

- a) SelectObject→Property
- b) FieldToUpdate→Verified
- c) FieldDataType→CheckBox
- d) SelectCheckBoxOptionas “True”
- e) Save.

SETUP **Field Updates**

Edit Field Update  
**Verified Property**

Define the field update, including the object associated with the workflow rule, approval process, or entitlement process, the field to update, and the value to apply. Note that the field to update may be on a related object. Fields are shown only for the type that you select.

**Field Update Edit**

Identification		* = Required Information
Name	<input type="text" value="Verified Property"/>	<input style="margin-left: 10px;" type="button" value="..."/>
Unique Name	<input type="text" value="Verified_Property"/> <input style="margin-left: 10px;" type="button" value="..."/>	<input style="margin-left: 10px;" type="button" value="..."/>
Description	<input type="text"/>	
Object	Property	
Field to Update	Property: Verified	
Field Data Type	Checkbox	
Re-evaluate Workflow Rules after Field Change	<input type="checkbox"/> <input style="margin-left: 10px;" type="button" value="..."/>	

**Specify New Field Value**

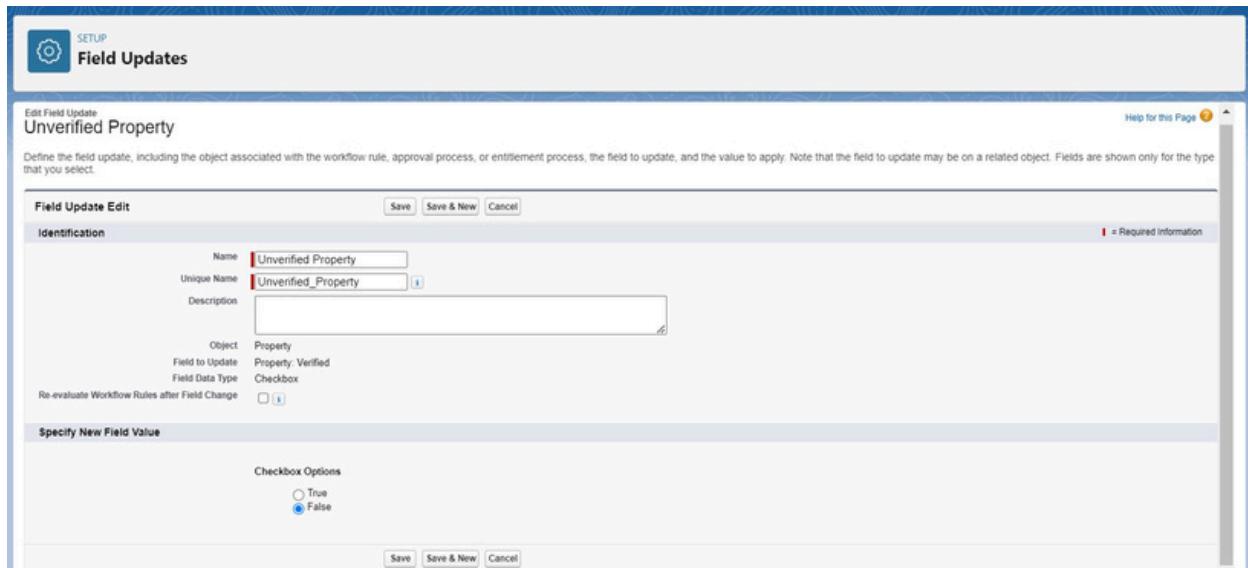
Checkbox Options

True  
 False

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

**13) Add One field Update as “UnVerified Property”**

- a) SelectObject→Property
- b) FieldtoUpdate→Verified
- c) FieldDataType→CheckBox
- d) SelectCheckBoxOptionas“False”
- e) Save.

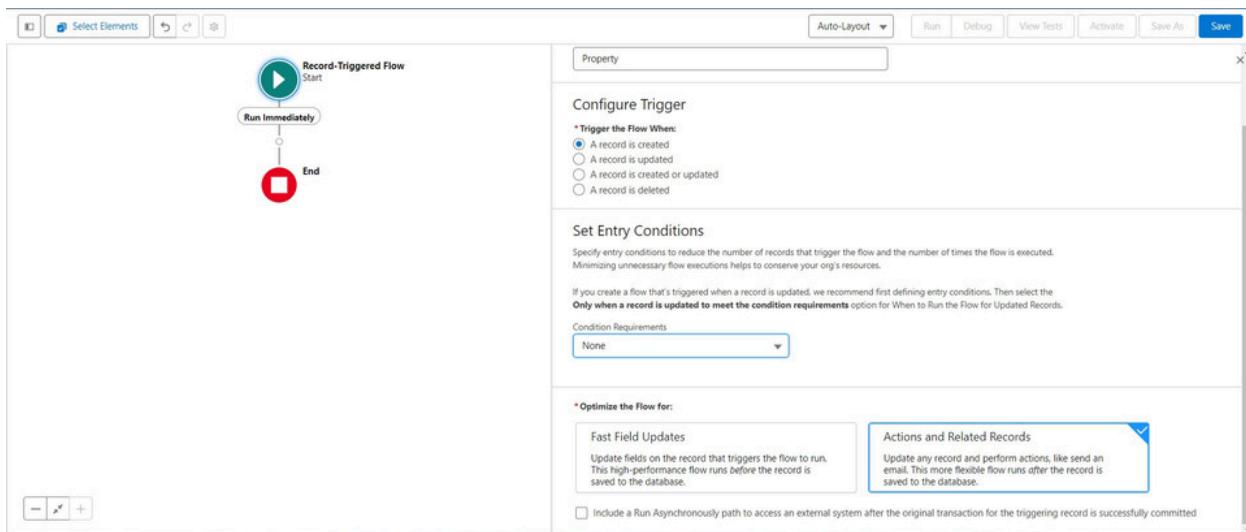


The screenshot shows the 'Field Updates' page in the Salesforce Setup. A new field update is being created for the 'Property' object. The 'Name' is set to 'Unverified Property' and the 'Unique Name' is 'Unverified\_Property'. The 'Field to Update' is 'Verified' and the 'Field Data Type' is 'Checkbox'. The 'Value' is set to 'False'. The 'Object' is 'Property'. The 'Re-evaluate Workflow Rules after Field Change' checkbox is unchecked. In the 'Specify New Field Value' section, the 'Checkbox Options' are set to 'False'. The page includes standard Salesforce navigation buttons like Save, Save & New, and Cancel.

**14) Activate the Approval Process.**

**Milestone 10 : - Create a Record trigger flow to submit the Approval Process Automatically.**

- 1) From Setup→Search for Flows→Click On New and Select“Record Trigger Flow”.
- 2) Select Object→Property
- 3) Select“Trigger the flow when”→“A record is created”
- 4) Set Entry Conditions→“None”



Record-Triggered Flow Start

Run Immediately

End

Configure Trigger

\* Trigger the Flow When:

- A record is created
- A record is updated
- A record is created or updated
- A record is deleted

Set Entry Conditions

Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.

If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the **Only when a record is updated to meet the condition requirements** option for When to Run the Flow for Updated Records.

Condition Requirements

None

\* Optimize the Flow for:

Fast Field Updates

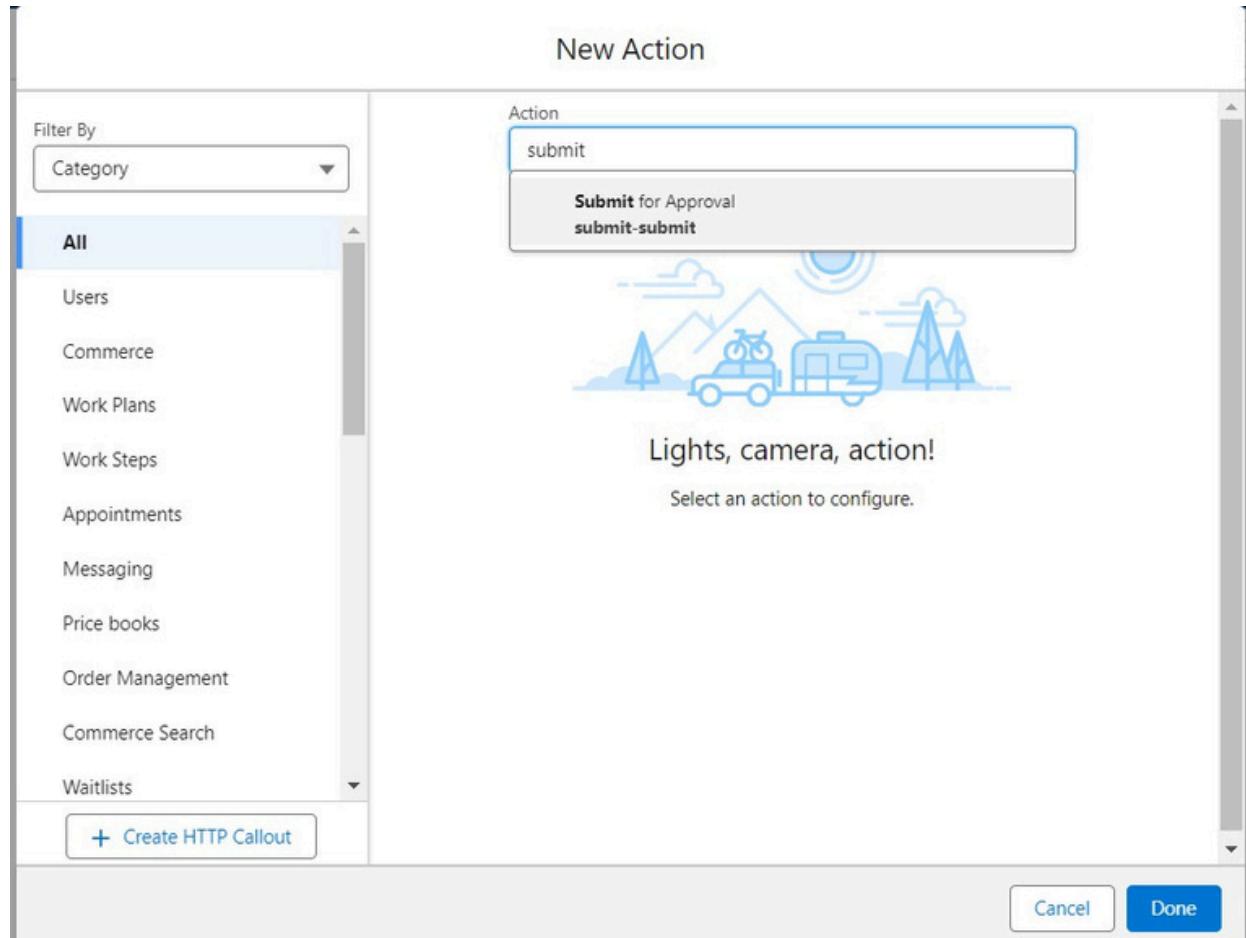
Update fields on the record that triggers the flow to run. This high-performance flow runs before the record is saved to the database.

Actions and Related Records

Update any record and perform actions, like send an email. This more flexible flow runs after the record is saved to the database.

Include a Run Asynchronously path to access an external system after the original transaction for the triggering record is successfully committed

## 5) Add a "Action" → "Submit for Approval"



New Action

Filter By

Category

All

Users

Commerce

Work Plans

Work Steps

Appointments

Messaging

Price books

Order Management

Commerce Search

Waitlists

+ Create HTTP Callout

Action

submit

Submit for Approval

submit-submit

Lights, camera, action!

Select an action to configure.

Cancel Done

6) Give Label→Approval for property

7) RecordId→{!\$Record.Id}

8) Done

### New Action

Filter By
Action

Category

**All**

- Users
- Commerce
- Work Plans
- Work Steps
- Appointments
- Messaging
- Price books
- Order Management
- Commerce Search
- Waitlists

Submit for Approval

Use values from earlier in the flow to set the inputs for the "Submit for Approval" core action. To use its outputs later in the flow, store them in variables.

\* Label

\* API Name

Description

Set Input Values for the Selected Action

<p>A_a * Record ID <input type="text" value="{\$Record.Id}"/></p> <hr/> <p>A_a Approval Process Name Or ID <input checked="checked" type="checkbox"/> Don't Include</p> <hr/> <p>A_a Next Approver IDs <input checked="checked" type="checkbox"/> Don't Include</p>
---

9) Save the Flow and Give label as→“PropertyApproval”and“Activate”

### Save the flow

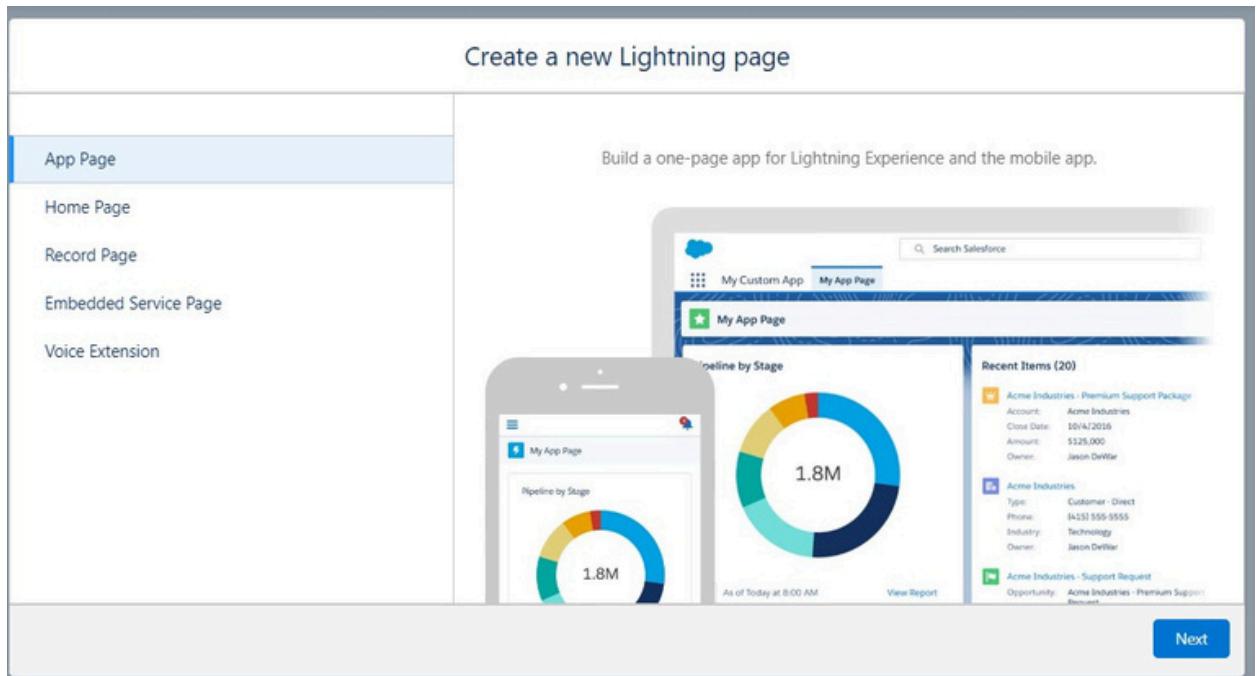
\* Flow Label
\* Flow API Name

Description

## Milestone 11 :- Create an App Page

- Create an App Page on the Property details Object named as “Search Your Property”

1) From Setup → Go to Lightning App Builder → Click on New → Select App Page and Click on Next.



- 2) Give Label as “Search your Property” click “Next”.
- 3) Click “header and Left Side bar” and Click on “Done”

## Create a new Lightning page

STANDARD (8)

Header and Left Sidebar

Header and Right Sidebar

Header and Three Regions

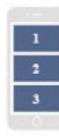
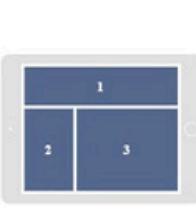
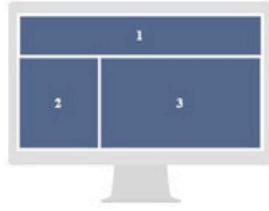
Header and Two Regions

Main Region and Right Sidebar

One Region

Three Regions

Two Regions

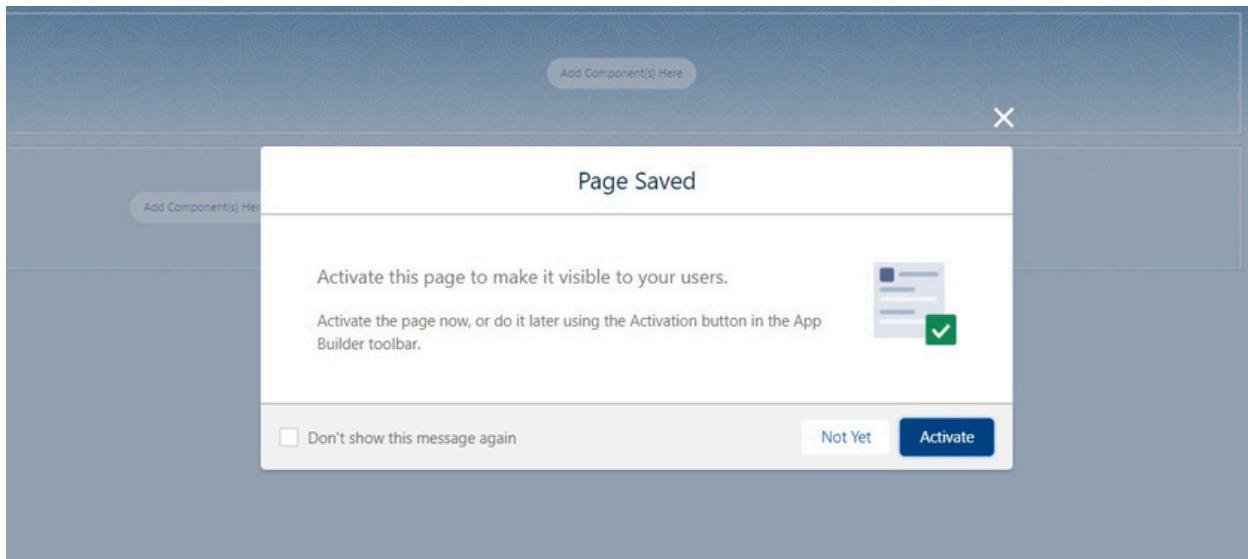


Full-width header above a left sidebar region and a wide main region. On a tablet in portrait orientation, the regions below the header are equal width. On a phone, the regions stack vertically.

**Supported form factors:** desktop, tablet, and phone.

[Back](#)[Done](#)

4) Click on “Save” and then click on “Activate”.



5) From Page Setting select page activation as “Activate for all Users”.

## Activation: Search your property

**PAGE SETTINGS**

## LIGHTNING EXPERIENCE

## MOBILE NAVIGATION

Give this app page a name, set the page visibility, and choose an icon.

**Name**

Enter a name for your page.

**Icon**

Choose an icon to represent your app in Lightning Experience and the mobile app.

[Change...](#)**Page Activation**

When you activate this page, a custom tab is created for it. You can manage the tab's visibility in Setup.

- Activate for all users  
 Activate for system administrators only

To set further restrictions on who sees this page, use permission sets and profile assignments in Setup.

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

## 6) From Lightning Experience Click on “Property Details” and click on Add Page“.

## Activation: Search your property

**PAGE SETTINGS****LIGHTNING EXPERIENCE****MOBILE NAVIGATION**

Add this app page to Lightning Experience apps. You can manage Lightning apps in Setup.

**Add to Lightning Apps**

	Lightning Bolt
	Lightning Instrumentation
	LWC Component
	Property Details
	Queue Management
	Sales
	Sales Console
	Salesforce CMS

**Property Details** [Remove page](#)

Search Your Property

Search your property

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

## 7) Then Click on “Save”

## TESTING AND VALIDATION

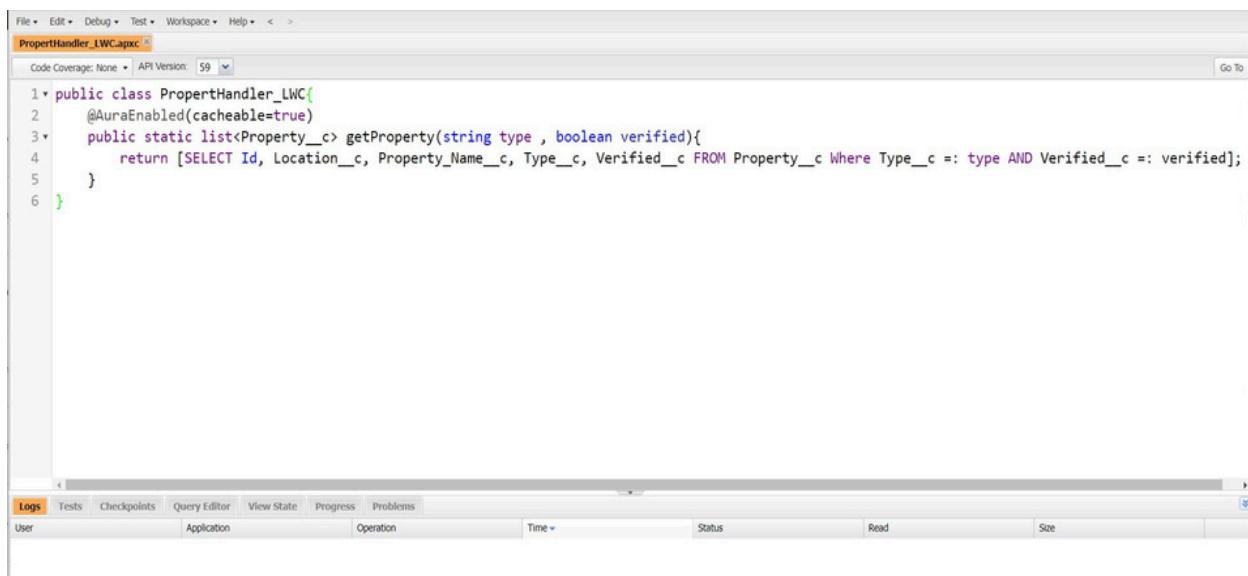
### Milestone 12 :- Create a LWC Component

- Create an Lwc Component for the customers so that only verified customers can access the verified properties and non Verified customers can access non verified properties, and deploy it on “Search your Property Page”

1) Create an Apex Class and make it aura enabled and name it “PropertHandler\_LWC”

Code:-

```
public class PropertHandler_LWC{
    @AuraEnabled(cacheable=true)
    public static list<Property__c> getProperty(string type , boolean verified){
        return [SELECT Id, Location__c, Property_Name__c, Type__c, Verified__c FROM
Property__c Where Type__c =: type AND Verified__c =: verified];
    }
}
```



The screenshot shows the Salesforce IDE interface with the code for the PropertHandler\_LWC apex class. The code is as follows:

```
File • Edit • Debug • Test • Workspace • Help • < >
PropertHandler_LWC.apex
Code Coverage: None | API Version: 59
1 * public class PropertHandler_LWC{
2     @AuraEnabled(cacheable=true)
3     public static list<Property__c> getProperty(string type , boolean verified){
4         return [SELECT Id, Location__c, Property_Name__c, Type__c, Verified__c FROM Property__c Where Type__c =: type AND Verified__c =: verified];
5     }
6 }
```

The code is syntax-highlighted, with keywords in green and identifiers in purple. The interface includes a toolbar at the top with options like File, Edit, Debug, Test, Workspace, and Help. Below the toolbar is a status bar showing 'Code Coverage: None' and 'API Version: 59'. At the bottom, there is a navigation bar with tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The 'Logs' tab is currently selected.

- 2) Create a Lightning Web Component in your VsCode, and (ctrl+shift+P) and click on authorize an org.

- 3) Enter your login id and password to authorize your org.
- 4) Now (ctrl+shift+P)→Create a Lightning Web Component and Name It Any thing you want to. (Example - )
- 5) In your Html File Write this code:-

Code :-

```
<template>
  <lightning-card>
    <div class="slds-box">
      <div class="slds-text-align_left">
        <h1 style="font-size: 20px;"><b>Properties</b></h1>
      </div>
      <div>
        <div class="slds-grid slds-gutters">
          <div class="slds-col slds-size_5-of-6">
            <lightning-combobox name="Type" label="Property Type" value={typevar}
placeholder="Select Property type"
options={propertyoptions} onchange={changehandler}></lightning-combobox>
          </div>
          <div class="slds-col slds-size_1-of-6">
            <br>
            <lightning-button-icon variant="neutral" icon-name="standard:search"
alternative-text="Search"
label="Search" onclick={handleClick}></lightning-button-icon>
          </div>
        </div>
      </div>
    </div>

  </div>

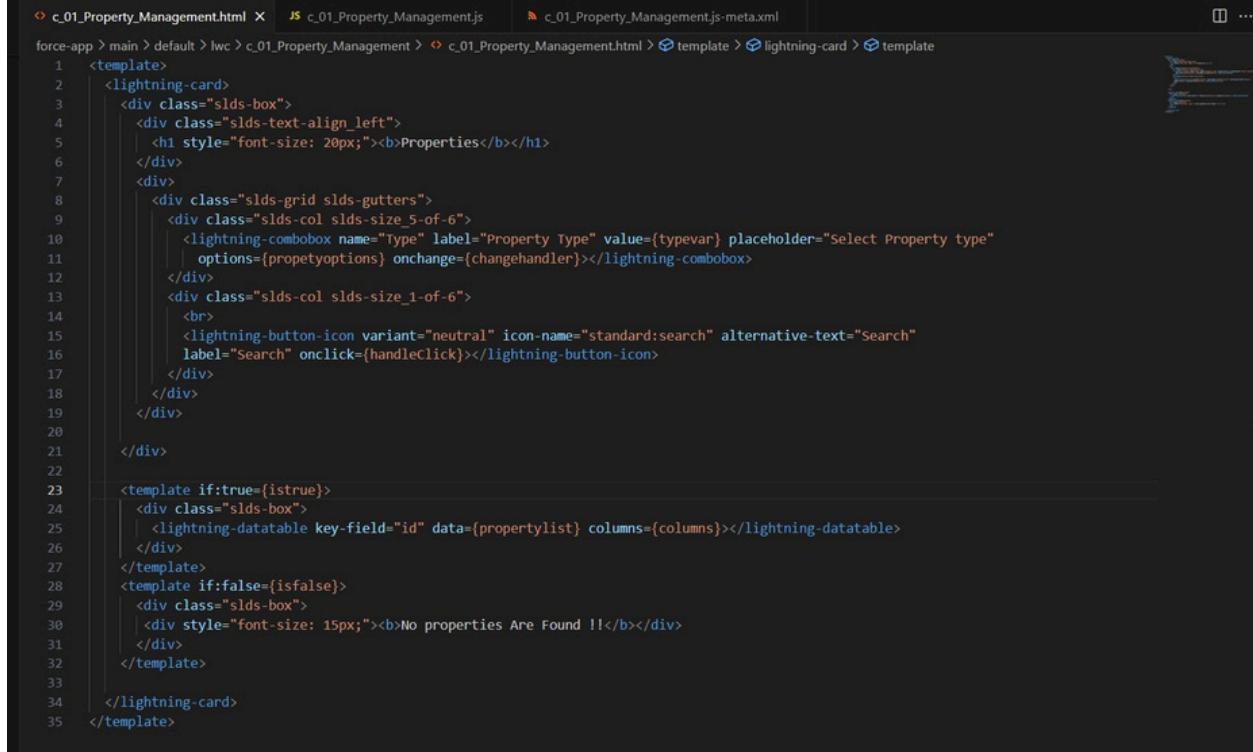
  <template if:true={istrue}>
    <div class="slds-box">
      <lightning-datatable key-field="id" data={propertylist}
columns={columns}></lightning-datatable>
    </div>
  </template>
  <template if:false={isfalse}>
```

```

<div class="slds-box">
  <div style="font-size: 15px;"><b>No properties Are Found !!</b></div>
</div>
</template>

</lightning-card>
</template>

```



```

c_01_Property_Management.html X JS c_01_Property_Management.js c_01_Property_Management.js-meta.xml
force-app > main > default > lwc > c_01_Property_Management > c_01_Property_Management.html > template > lightning-card > template
  1  <template>
  2    <lightning-card>
  3      <div class="slds-box">
  4        <div class="slds-text-align_left">
  5          <h1 style="font-size: 20px;"><b>Properties</b></h1>
  6        </div>
  7        <div>
  8          <div class="slds-grid slds-gutters">
  9            <div class="slds-col slds-size_5-of-6">
 10              <lightning-combobox name="Type" label="Property Type" value={typevar} placeholder="Select Property type"
 11                options={propertysoptions} onchange={changehandler}></lightning-combobox>
 12            </div>
 13            <div class="slds-col slds-size_1-of-6">
 14              <br>
 15              <lightning-button-icon variant="neutral" icon-name="standard:search" alternative-text="Search"
 16                label="Search" onclick={handleClick}></lightning-button-icon>
 17            </div>
 18          </div>
 19        </div>
 20      </div>
 21
 22      <template if:true={isttrue}>
 23        <div class="slds-box">
 24          <lightning-datatable key-field="id" data={propertylist} columns={columns}></lightning-datatable>
 25        </div>
 26      </template>
 27      <template if:false={isfalse}>
 28        <div class="slds-box">
 29          <div style="font-size: 15px;"><b>No properties Are Found !!</b></div>
 30        </div>
 31      </template>
 32    </div>
 33  </template>
 34 </lightning-card>
 35 </template>

```

6) In Your Js File Write this code :-

Code :-

```

import { LightningElement, api, track, wire } from 'lwc';
import getProperty from "@salesforce/apex/PropertHandler_LWC.getProperty"
import { getRecord } from 'lightning/uiRecordApi';
import USER_ID from '@salesforce/user/Id';
export default class C_01_Property_Management extends LightningElement {
  @api recordId
  userId = USER_ID;

```

```
verifiedvar      typevar
isfalse = true; istrue =
false; @track propertylist
= []; columns = [
]

{ label: 'Property Name', fieldName: 'Property_Name__c' },
{ label: 'Property Type', fieldName: 'Type__c' },
{ label: 'Property Location', fieldName: 'Location__c' },
{ label: "Property link", fieldName: "Property_link__c" }
]
propertyoptions = [
    { label: "Commercial", value: "Commercial" },
    { label: "Residential", value: "Residential" },
    { label: "rental", value: "rental" }
]

]
@wire(getRecord, { recordId: "$userId", fields: ['User.Verified__c'] })
recordFunction({ data, error }) {
    if (data) {
        console.log(data)
        console.log("This is the User Id ---> "+this.userId);
        this.verifiedvar = data.fields.Verified__c.value;
    } else {
        console.error(error)
        console.log('this is error')
    }
}

changehandler(event) {
    console.log(event.target.value);
    this.typevar = event.target.value;
}
handleClick() {

    getProperty({ type: this.typevar, verified: this.verifiedvar })
        .then((result) => {
            this.isfalse = true;
            console.log(result)
            console.log('This is the User id ---> ' + this.userId);
        })
}
```

```
        console.log('This is the verified values ---> ' + this.verifiedvar);
        if (result != null && result.length != 0) {
            this.isTrue = true;
            this.propertylist = result;
            console.log(this.verifiedvar);
            console.log(this.typevar)
        } else {
            this.isFalse = false;
            this.isTrue = false;
        }

    })
    .catch((error) => {
        console.log(error)
    })
}
```

```

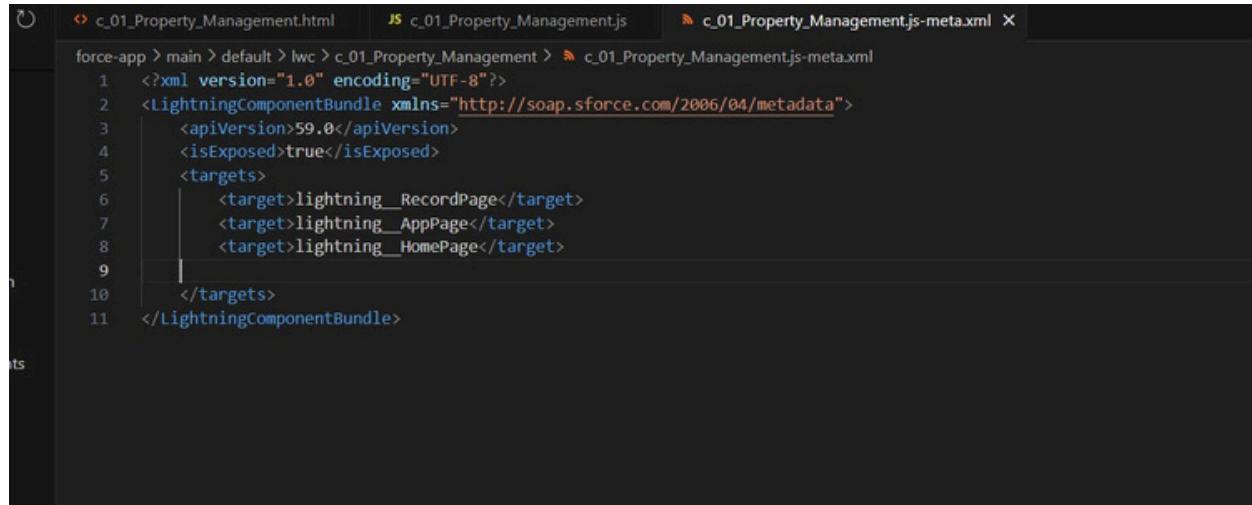
c_01_Property_Management.html JS c_01_Property_Management.js X c_01_Property_Management.js-meta.xml
force-app > main > default > lwc > c_01_Property_Management > JS c_01_Property_Management.js > c_01_Property_Management > propertyoptions
1 import { LightningElement, api, track, wire } from 'lwc';
2 import getProperty from "@salesforce/apex/Properthandler_LWC.getProperty"
3 import { getRecord } from 'lightning/uiRecordApi';
4 import USER_ID from '@salesforce/user/Id';
5 export default class C_01_Property_Management extends LightningElement {
6     @api recordId;
7     userId = USER_ID;
8     verifiedvar;
9     typevar;
10    isfalse = true;
11    istrue = false;
12    @track propertylist = [];
13    columns = [
14        { label: 'Property Name', fieldName: 'Property_Name_c' },
15        { label: 'Property Type', fieldName: 'Type_c' },
16        { label: 'Property Location', fieldName: 'Location_c' },
17        { label: "Property link", fieldName: "Property_link_c" }
18    ];
19    propertyoptions = [
20        { label: "Commercial", value: "Commercial" },
21        { label: "Residential", value: "Residential" },
22        { label: "rental", value: "rental" }
23    ];
24    @wire(getRecord, { recordId: "$userId", fields: ['User.Verified_c'] })
25    recordFunction({ data, error }) {
26        if (data) {
27            console.log(data);
28            console.log("This is the User Id ---> " + this.userId);
29            this.verifiedvar = data.fields.Verified_c.value;
30        } else {
31            console.error(error);
32            console.log('this is error');
33        }
34    }
35}
36
37    changehandler(event) {
38        console.log(event.target.value);
39        this.typevar = event.target.value;
40    }
41    handleClick() {
42
43        getProperty({ type: this.typevar, verified: this.verifiedvar })
44        .then((result) => {
45            this.isfalse = true;
46            console.log(result);
47            console.log('This is the User id ---> ' + this.userId);
48            console.log('This is the verified values ---> ' + this.verifiedvar);
49            if (result != null && result.length != 0) {
50                this.istrue = true;
51                this.propertylist = result;
52                console.log(this.verifiedvar);
53                console.log(this.typevar)
54            } else {
55                this.isfalse = false;
56                this.istrue = false;
57            }
58        })
59        .catch((error) => {
60            console.log(error)
61        })
62    }
63}
64
65
66}

```

7) In Your metafile give your targets to deploy the component.

Code :-

```
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
<apiVersion>59.0</apiVersion>
<isExposed>true</isExposed>
<targets>
<target>lightning__RecordPage</target>
<target>lightning__AppPage</target>
<target>lightning__HomePage</target>
</targets>
</LightningComponentBundle>
```



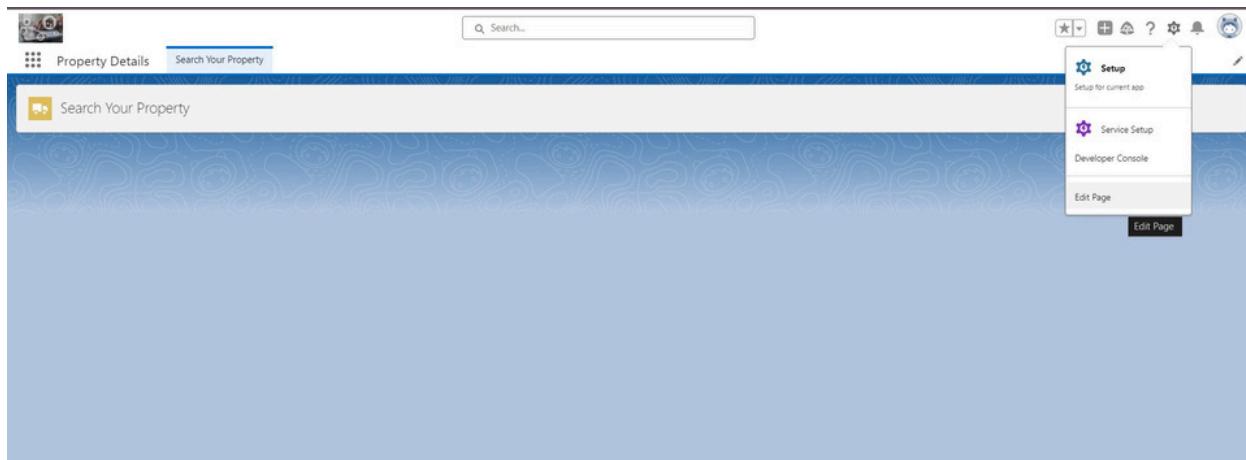
```
force-app > main > default > lwc > c_01_Property_Management > c_01_Property_Management.js-meta.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
3    <apiVersion>59.0</apiVersion>
4    <isExposed>true</isExposed>
5    <targets>
6      <target>lightning__RecordPage</target>
7      <target>lightning__AppPage</target>
8      <target>lightning__HomePage</target>
9    </targets>
10   </LightningComponentBundle>
```

8) After Saving all the three Codes, Right Click and Deploy this component to the org.

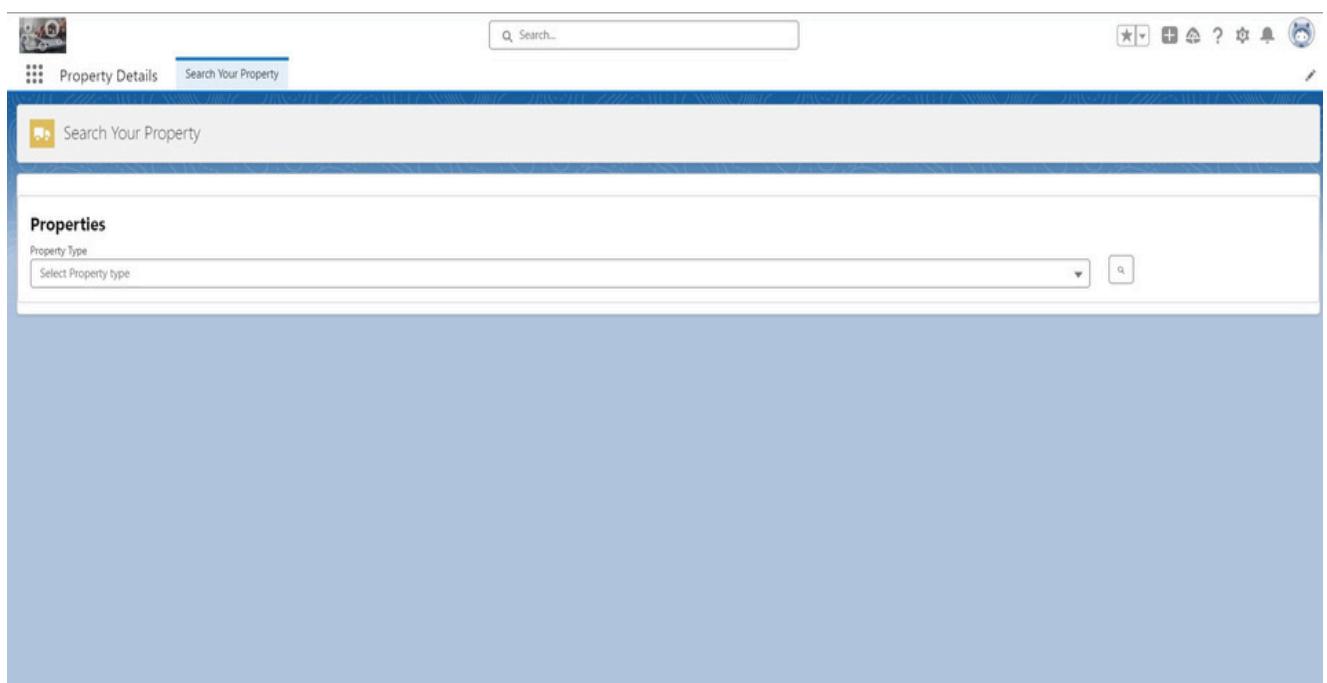
## Milestone 13 : - Drag this Component to your App Page

1) From Setup → Go to App Launcher → Search for Property Details

2) On this Page click on gear icon and click on Edit Page



3) Drag the Component to your App Page and Save the Page.



## Milestone 14 : - Give Access of Apex Classes to Profiles

- 1) From Setup→Search For Apex Classes→Click on “Security” behind “Property Handler\_\_LWC”.
- 2) From Profiles Add “Manager” and “Customer” and “Save”.

## Conclusion:

A CRM application for managing clients and property-related requirements streamlines operations, enhances client communication, and improves property tracking.

By centralizing data, automating lead management, and offering insightful reporting, it boosts efficiency and strengthens client relationships, ultimately driving better business outcomes in real estate and property management.