A CRM Application to Handle the Clients and their property Related Requirements (PropertyCRM):

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.

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

- **Create a Jotform Account:**
    - **If you don't already have a Jotform account, sign up for one on the Jotform website.**
- **Start a New Form:**
    - **Once logged in, click on the "Create Form" button.**
    - **Choose to either start from scratch or use a template.**
- **Add Elements:**
    - **When starting from scratch, an empty form will appear.**
    - **Begin adding form elements like text fields, checkboxes, radio buttons, etc., as per your requirements.**

    - You can find these elements in the Jotform Builder interface.

- **Set Required Fields:**
    - For any field that should be mandatory, mark it as required by checking the "Required" option. This will display a "*" symbol next to it, ensuring the user must fill it out before submitting the form.

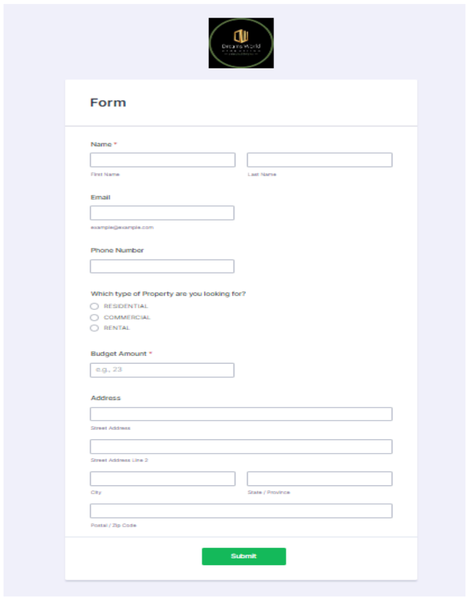
- **Review & Correct:**
    - Double-check the form for errors, such as missing required fields, incorrect labels, or layout issues.
    - Correct any errors you find before publishing.

- **Publish the Form:**
    - Once everything looks good, click the "Publish" button.
    - This will make the form publicly available, and you will be provided with a link to the form.

- **Test the Form:**
    - After publishing, use the provided link to test the form. Identify any further errors or areas for improvement and correct them by going back to the form editor.

JOTFORM LINK

https://form.jotform.com/242751789204058



Form

Name *

First Name | Last Name

Email

example@example.com

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

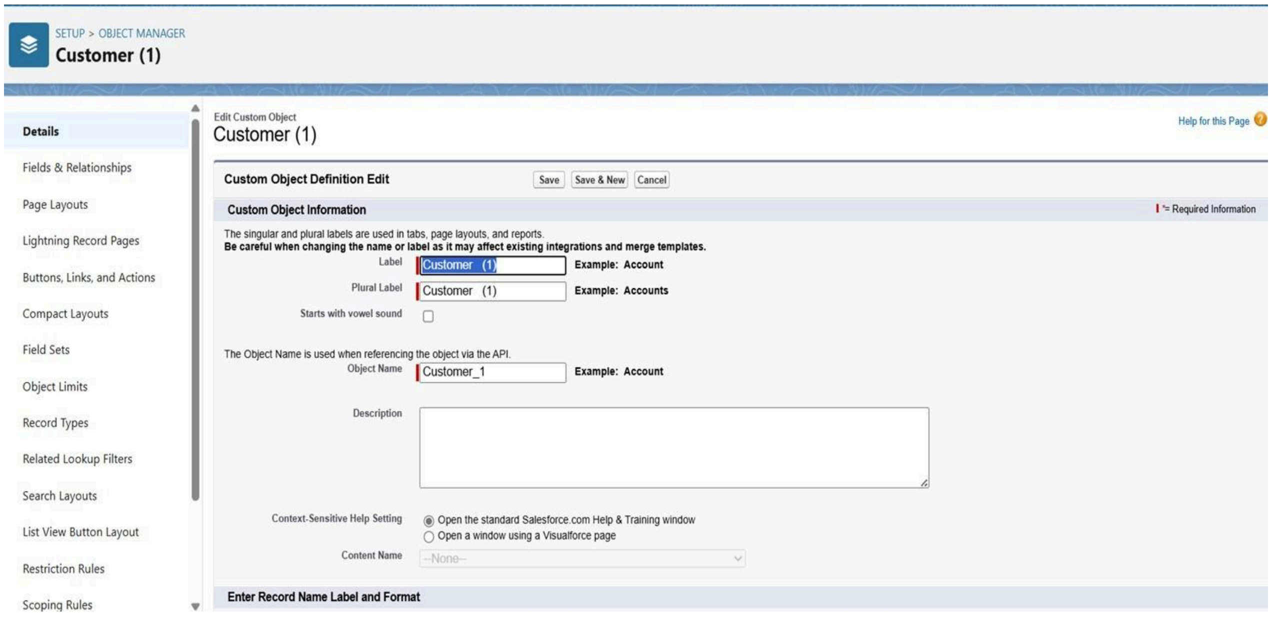
City | State / Province

Postal / Zip Code

Submit

Create Objects from Spreadsheet.

Directly Creating Objects from Spreadsheet in Salesforce



Edit Custom Object
Customer (1)

Custom Object Definition Edit

Custom Object Information

| = Required Information

The singular and plural labels are used in tabs, page layouts, and reports.
Be careful when changing the name or label as it may affect existing integrations and merge templates.

Label	Customer (1)	Example: Account
Plural Label	Customer (1)	Example: Accounts
Starts with vowel sound	☐	

The Object Name is used when referencing the object via the API.

Object Name	Customer_1	Example: Account
Description		

Context-Sensitive Help Setting: [✓ Open the standard Salesforce.com Help & Training window] Open a window using a Visualforce page

Content Name: --None--

Enter Record Name Label and Format

Steps to Create a Custom Object Named "Customer" in Salesforce

1. **Log in to Salesforce:**
    - **Navigate to your Salesforce instance and log in with your credentials.**
2. **Go to Setup:**
    - **Click on the gear icon (◦Oⓧ) in the top-right corner.**
    - **Select Setup from the dropdown.**
3. **Navigate to Object Manager:**
    - **In the Setup menu, type "Object Manager" in the Quick Find box.**
    - **Click on Object Manager.**

4. **Create a New Custom Object:**

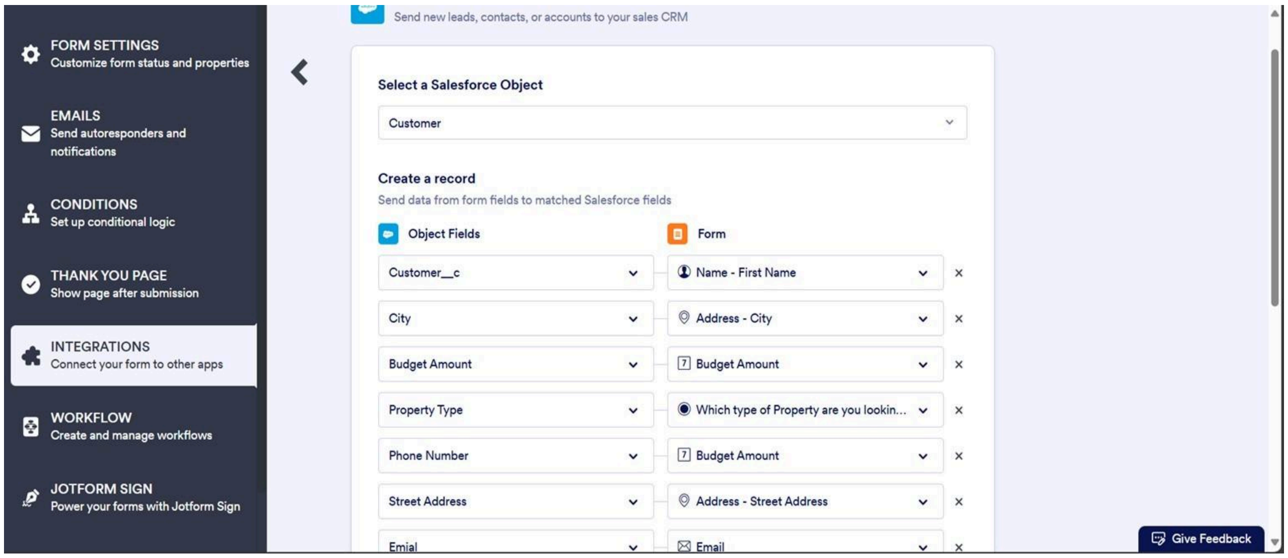
   - **In the Object Manager, click the Create dropdown in the top-right corner and select Custom Object.**

5. **Define the Custom Object Properties:**

   - **Label: Enter "Customer" (this will appear in the UI).**
   - **Plural Label: Enter "Customers" (for list views, reports, etc.).**
   - **Object Name: Salesforce will automatically generate an API Name, but you can edit it if necessary (e.g., Customer_c).**
   - **Record Name: This will be the primary field for the object. For example, you can choose "Customer Name" and set it as either a text field or an auto-number.**

SAME PROCESS APPLIED FOR CREATING PROPERTY OBJECT

Integrate Jotform with Salesforce Platform


FORM SETTINGS
Customize form status and properties
EMAILS
Send autoresponders and notifications
CONDITIONS
Set up conditional logic
THANK YOU PAGE
Show page after submission
INTEGRATIONS
Connect your form to other apps
WORKFLOW
Create and manage workflows
JOTFORM SIGN
Power your forms with Jotform Sign
Give Feedback

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

Select a Salesforce Object

Customer

Create a record

Send data from form fields to matched Salesforce fields

Object Fields	Form
Customer__c	Name - First Name
City	Address - City
Budget Amount	Budget Amount
Property Type	Which type of Property are you lookin...
Phone Number	Budget Amount
Street Address	Address - Street Address
Emial	Email

Steps to Integrate Jotform with Salesforce:

1. **Create a Jotform Account and Form**

- **If you haven't already, create a Jotform account and design your form. If you need help**

creating a form, you can follow the steps outlined earlier.

2. **Connect Jotform to Salesforce**

- **Log into Jotform: After logging into your Jotform account, go to the form you want to integrate with Salesforce.**
- **Go to Settings:**
    - ▪ **Open your form in the Form Builder.**
    - ▪ **Click on the Settings tab at the top.**
- **Integrations:**
    - ▪ **In the left sidebar, click on Integrations.**
    - ▪ **In the search bar, type Salesforce and click on the Salesforce icon.**

3. **Authenticate Salesforce**

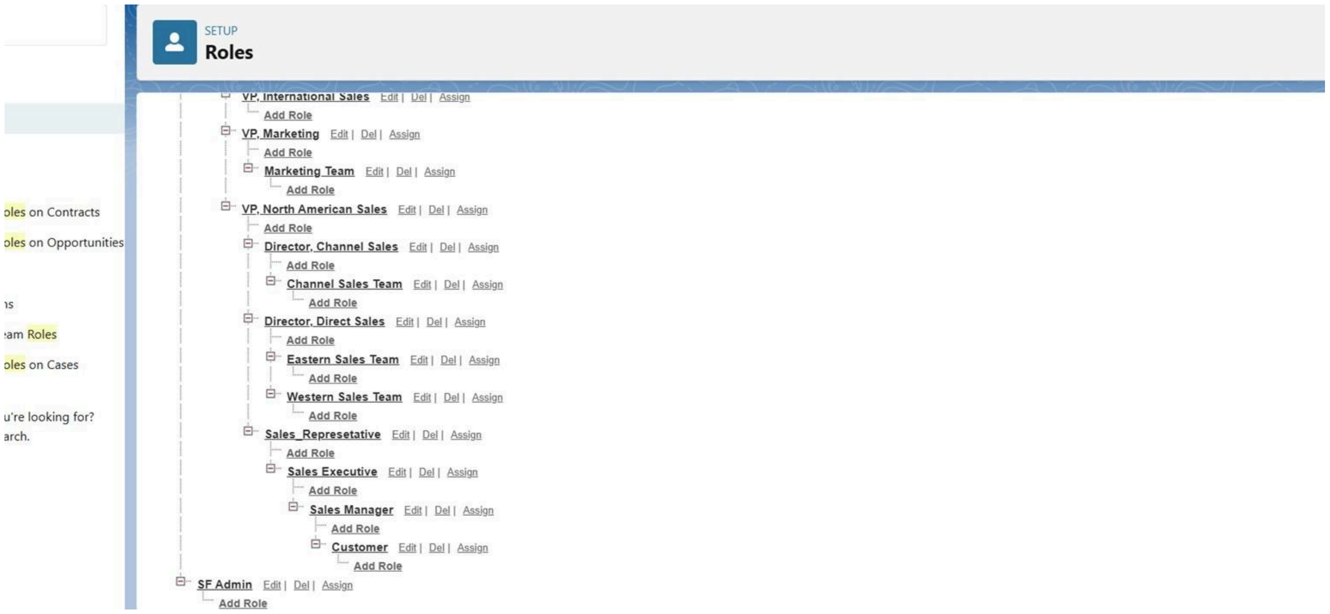
- **Jotform will prompt you to log into Salesforce.**
- **Enter your Salesforce credentials and allow Jotform to access your Salesforce account.**

4. **Configure the Integration**

- **After successfully logging in, you will need to map the Jotform fields to Salesforce objects and fields.**
- **Choose Salesforce Object:**
    - ▪ **Jotform will ask you to select which Salesforce object to connect the form to (e.g., Leads, Contacts, Opportunities, Custom Objects like the "Customer" object you created earlier).**
- **Map Jotform Fields to Salesforce Fields:**

- **Jotform will display a list of fields from your form and the selected Salesforce object. For each form field, choose the corresponding Salesforce field to map it to.**
- **For example, if your form has fields for "Name," "Email," and "Phone Number," you can map these to the corresponding fields in Salesforce (like Contact Name, Email, Phone).**

Create Roles



```
VP, International Sales  Edit | Del | Assign
  Add Role
VP, Marketing  Edit | Del | Assign
  Add Role
  Marketing Team  Edit | Del | Assign
    Add Role
VP, North American Sales  Edit | Del | Assign
  Add Role
  Director, Channel Sales  Edit | Del | Assign
    Add Role
    Channel Sales Team  Edit | Del | Assign
      Add Role
  Director, Direct Sales  Edit | Del | Assign
    Add Role
    Eastern Sales Team  Edit | Del | Assign
      Add Role
    Western Sales Team  Edit | Del | Assign
      Add Role
  Sales_Represetative  Edit | Del | Assign
    Add Role
    Sales Executive  Edit | Del | Assign
      Add Role
      Sales Manager  Edit | Del | Assign
        Add Role
        Customer  Edit | Del | Assign
          Add Role
SF Admin  Edit | Del | Assign
  Add Role
```

- **Navigate to Role Settings.**

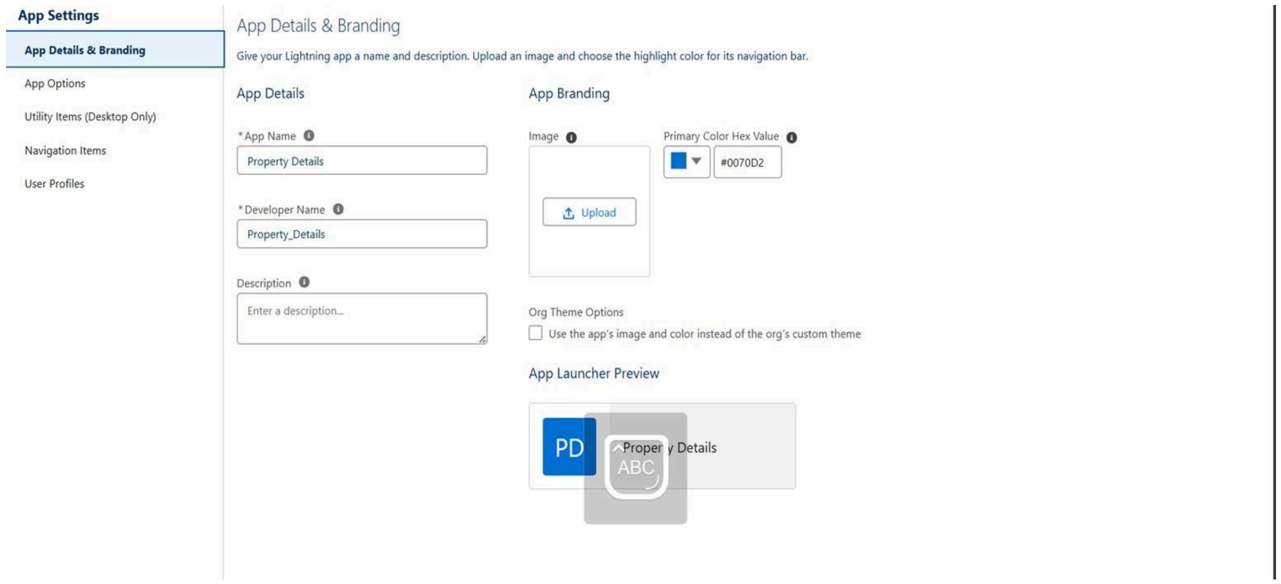
- **In the Quick Find search bar (on the left sidebar), type Roles.**
- **Click on Roles under the Users section.**
- **Set Up Roles Hierarchy.**
- **You will be directed to the Role Hierarchy page, where you can see the default roles that exist in Salesforce.**
- **Click the Set Up Roles button if you are setting up roles for the first time.**
- **Create a New Role.**

- To create a new role, click the Add Role link next to an existing role, or if starting from scratch, click Add Role at the top of the hierarchy tree.

- **Define Role Details.**
- **Label: Enter a name for the role (e.g., "Sales Manager," "Customer Support Agent").**
- **Role Name: Salesforce will auto-generate a role name based on the label you provide.**
- **This role reports to: Select the role to which this new role will report. For example, if**

you're creating a "Sales Manager" role, it might report to "VP of Sales."

Create a Property Details App


App Settings
App Details & Branding
App Options
Utility Items (Desktop Only)
Navigation Items
User Profiles
App Details & Branding
Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.
App Details
App Branding
Image
Upload
Org Theme Options
Use the app's image and color instead of the org's custom theme
App Launcher Preview
PD
Property Details

App Name: Property Details
Developer Name: Property_Details
Description: Enter a description...
Primary Color Hex Value: #0070D2

- **Create a Custom Object for Property.**
- **In the Setup search bar (on the left sidebar), type Object Manager and click on it.**
    - **Click the Create button and select Custom Object.**

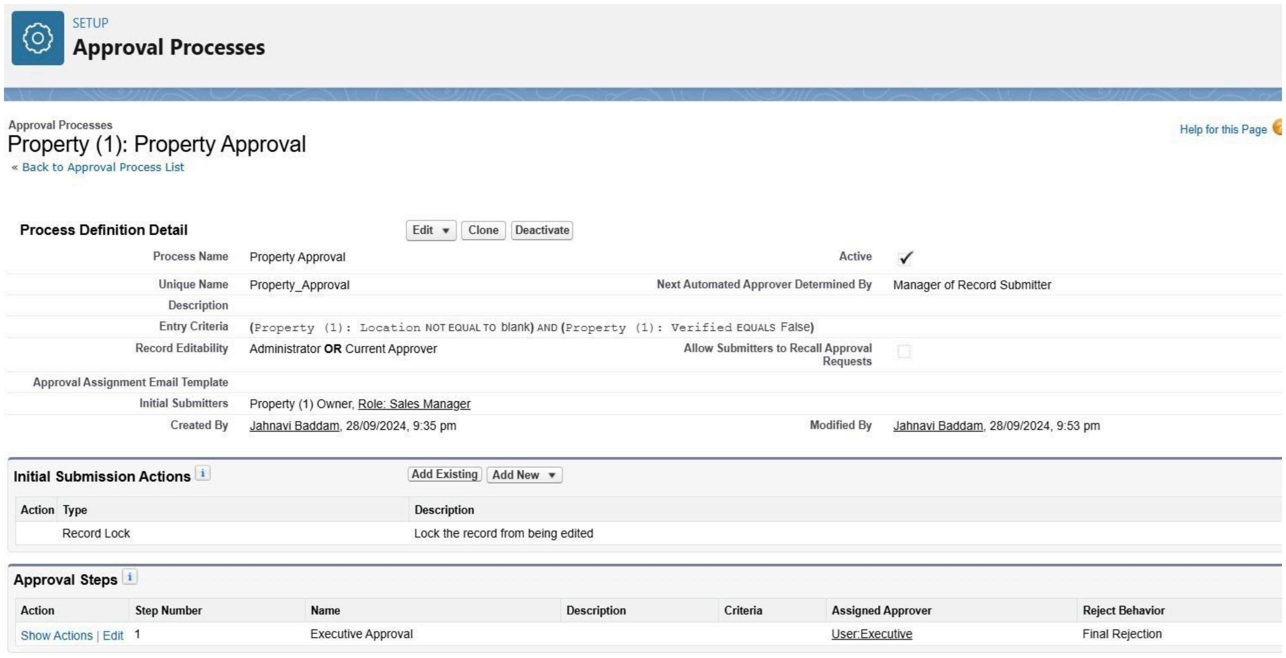
Define the custom object:

- **Label: Enter "Property".**

- **Plural Label: Enter "Properties".**
- **Object Name: Salesforce will auto-generate an API Name (e.g., Property_c).**
- **Record Name: Choose "Property Name" and select Text.**
- **Check the boxes for Allow Reports, Allow Activities, and Track Field History if needed.**
- **Click Save.**
- **Add Custom Fields to the Property Object.**
- **Click Save.**
- **Now, add fields to the Property object to capture details such as price, size, location, etc.**
- **Go to the newly created Property object.**

- **Click Fields & Relationships, then click New to create new fields.**

Create an Approval Process for Property Object



Property (1): Property Approval

« Back to Approval Process List

Process Definition Detail

Process Name	Property Approval	Active	✓
Unique Name	Property_Approval	Next Automated Approver Determined By	Manager of Record Submitter
Description			
Entry Criteria	(Property (1): Location NOT EQUAL TO blank) AND (Property (1): Verified EQUALS False)		
Record Editability	Administrator **OR** Current Approver	Allow Submitters to Recall Approval Requests	☐
Approval Assignment Email Template			
Initial Submitters	Property (1) Owner, Role: Sales Manager		
Created By	Jahnavi Baddam, 28/09/2024, 9:35 pm	Modified By	Jahnavi Baddam, 28/09/2024, 9:53 pm

Initial Submission Actions

Action	Type	Description
	Record Lock	Lock the record from being edited

Approval Steps

Action	Step Number	Name	Description	Criteria	Assigned Approver	Reject Behavior
Show Actions | Edit	1	Executive Approval			User:Executive	Final Rejection

    - Navigate to Approval Processes.
    - In the Quick Find box (on the left sidebar), type Approval Processes and click on it under Process Automation
    - Select Property Object.

- Under the Approval Processes section, click Create New Approval Process and choose Use Standard Setup Wizard.
- In the dropdown, select Property as the object you want the approval process to apply to.
- Define the Approval Process.
- Initial Setup:
    - Process Name: Enter a name for your approval process (e.g., "Property Listing Approval").
- Unique Name: Salesforce will auto-generate this field, but you can edit it if needed.
- Entry Criteria: Specify the criteria that must be met for the approval process to be triggered.
        - Example: You could set the criteria to trigger the approval process only when the Status of the property is Pending Approval.
        - Use Filter Criteria like:
            - Property.Status = "Pending Approval"
    - Record Editability: Choose who can edit the records during the approval process. Select whether users can edit the record only when it's assigned to them or if they can always edit it.
- Click Next.
- Select Approver and Configure Actions.
- Step 1: Select Approval Assignments.
    - Submitter Type: Choose who can submit a record for approval (e.g., "Only Record Owner," "Manager," or "Anyone").
- Next Approver: Choose how the next approver is assigned. You can assign it to a specific user, role, or even a queue (e.g., "Sales Manager" or "VP of Real Estate").

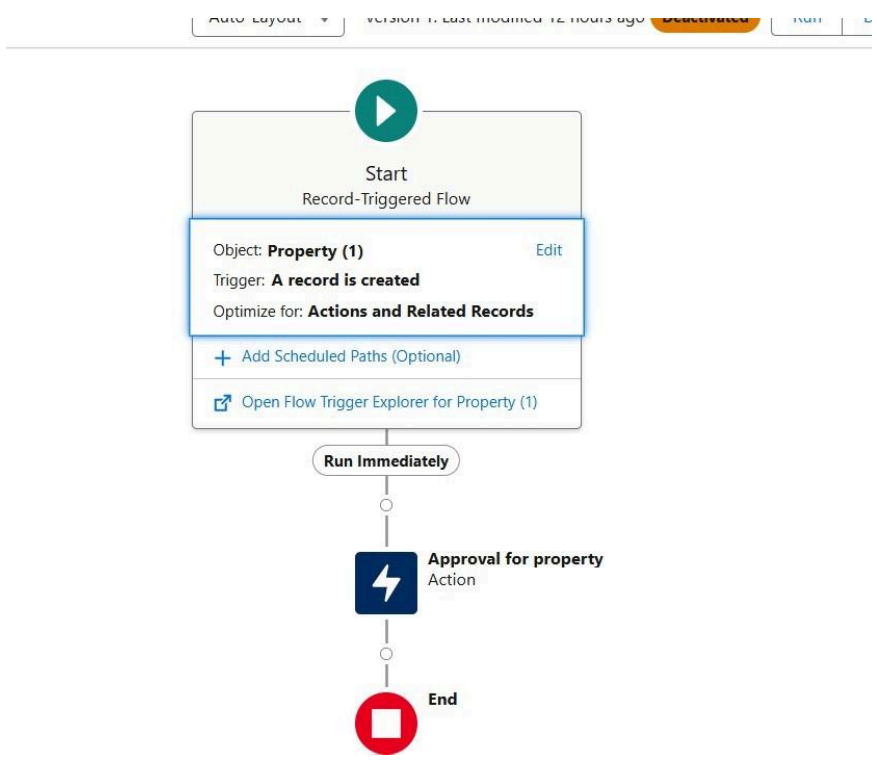
        - Example: Assign the approval request to the user's manager or a specific role such as Property Manager.

- Click Next.
- Specify Initial Submission Actions.
    - Define what happens when a record is first submitted for

approval. Actions to configure:

- Email Alert: Send an email alert to the approver(s) notifying them of the pending approval.
        - You can create a new email template and select it here.
- Field Update: Automatically update the property status to "Verified property" and "Unverified Property"

Create a Record trigger flow to submit the Approval Process Automatically.



Object: **Property (1)**
Trigger: **A record is created**
Optimize for: **Actions and Related Records**

Edit
Add Scheduled Paths (Optional)
Open Flow Trigger Explorer for Property (1)
Run Immediately
Approval for property
Action
End

- **Navigate to Flow Builder**
- **In the Quick Find box (on the left sidebar), type Flows and select Flows under Process Automation.**
- **Click New Flow to create a new flow. Select Flow Type**
- **Choose Record-Triggered Flow.**
- **Click Next.**
- **Define Flow Trigger.**
- **Object: Select Property (the custom object you created for property listings).**

- **Trigger the Flow When: Select A record is created or updated.**
  - **Condition Requirements: Select Condition is met to specify when the flow should trigger.**

Set Action Details:

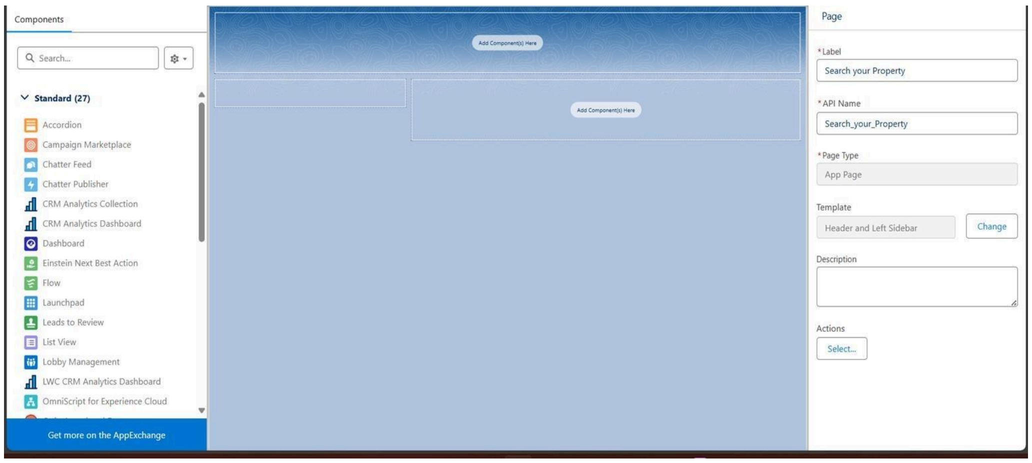
- **Action Type: In the search box, type Submit for Approval.**
  - **Select Submit for Approval from the**

list. Set Parameters for the Approval

Submission:

- **Record ID: In the Record ID field, insert the Property's record ID. You can choose this from the variable options.**
  - **In the Flow Builder, you'll often see an option like $Record.Id representing the Property record's ID.**
- **Approval Process Name: If you have multiple approval processes for the Property object, you can select the specific approval process to use here. If not specified, Salesforce will use the default approval process for the Property object.**

Create an App Page



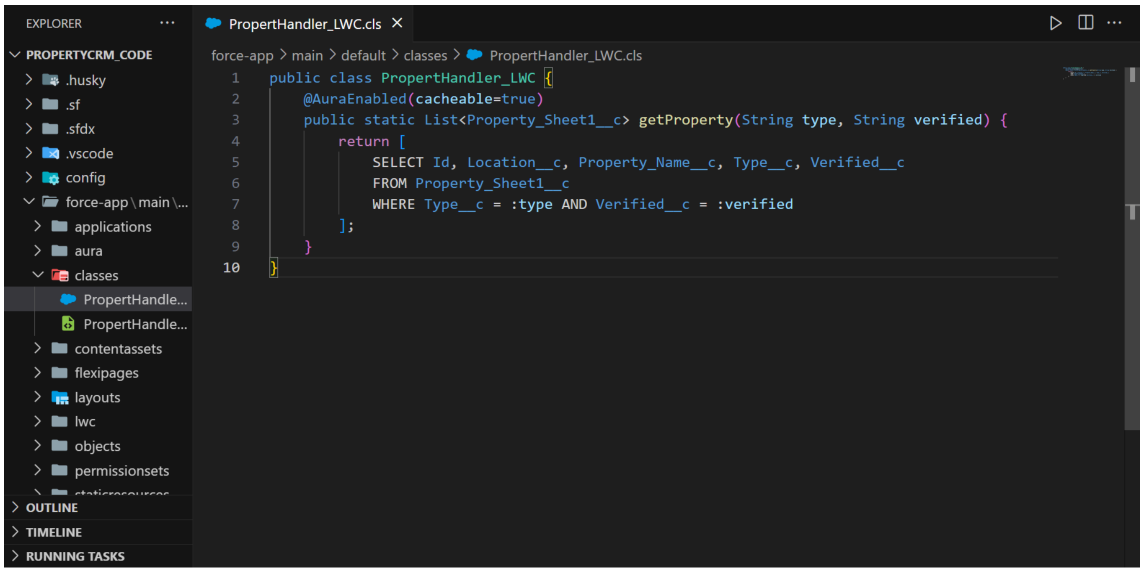
*Label: Search your Property
*API Name: Search_your_Property
*Page Type: App Page
Template: Header and Left Sidebar
Description:
Actions:

- **Go to Lightning App Builder**
  - **In the Quick Find search bar, type Lightning App Builder and select it from the results.**

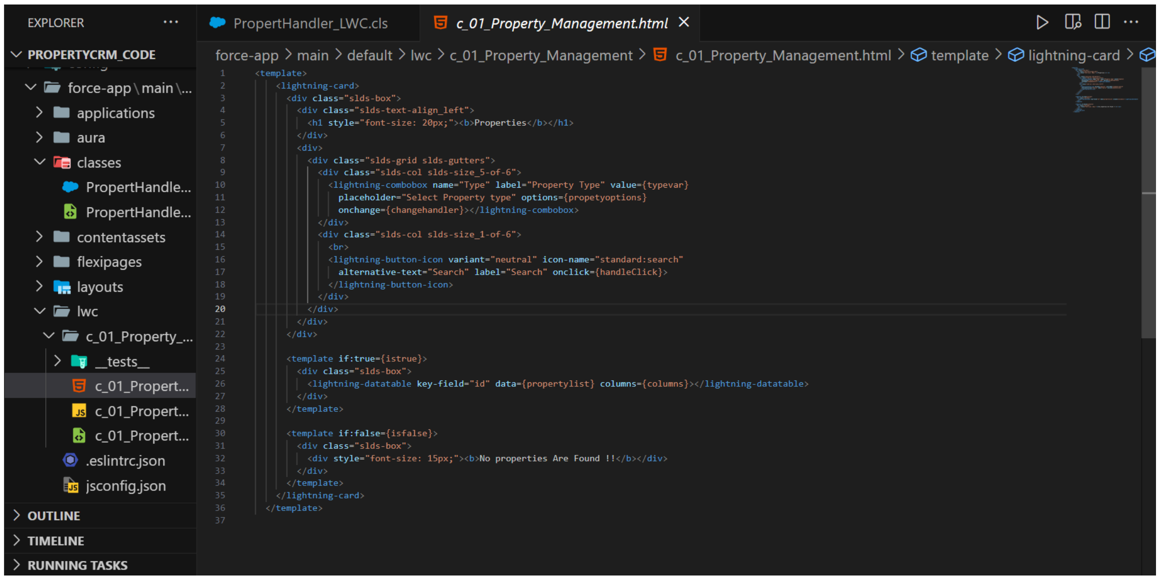
- **Create a New App Page**
    - **In the Lightning App Builder, click the New button.**
- **Select App Page Type**
    - **Choose App Page from the available options.**
    - **Click Next.**
- **Label the App Page**
    - **In the Label field, enter “Search Your Property”.**
    - **Click Next.**
- **Choose Page Layout**
    - **Select the layout option Header and Left Sidebar from the available layout options.**
    - **Click Done.**
- **Customize the App Page (Optional)**
    - **You can now customize the page by dragging and dropping components (like search bars, list views, etc.) into the header or left sidebar area, depending on your requirements.**
- **For this basic setup, you can skip adding any components for now if you're just setting up the page structure.**
- **Save the Page**
    - **Once your layout is configured, click Save in the top-right corner of the screen.**
- **Activate the App Page**
    - **After saving, click the Activate button.**
- **Page Activation Settings**
    - **You will be prompted with the Page Activation Settings. Here, select Activate for All Users to make the app page available to everyone.**
    - **You can choose to activate the page for specific profiles or apps if needed, but to follow your steps, select the option for all users.**

- **Click Next or Finish after selecting the activation setting.**

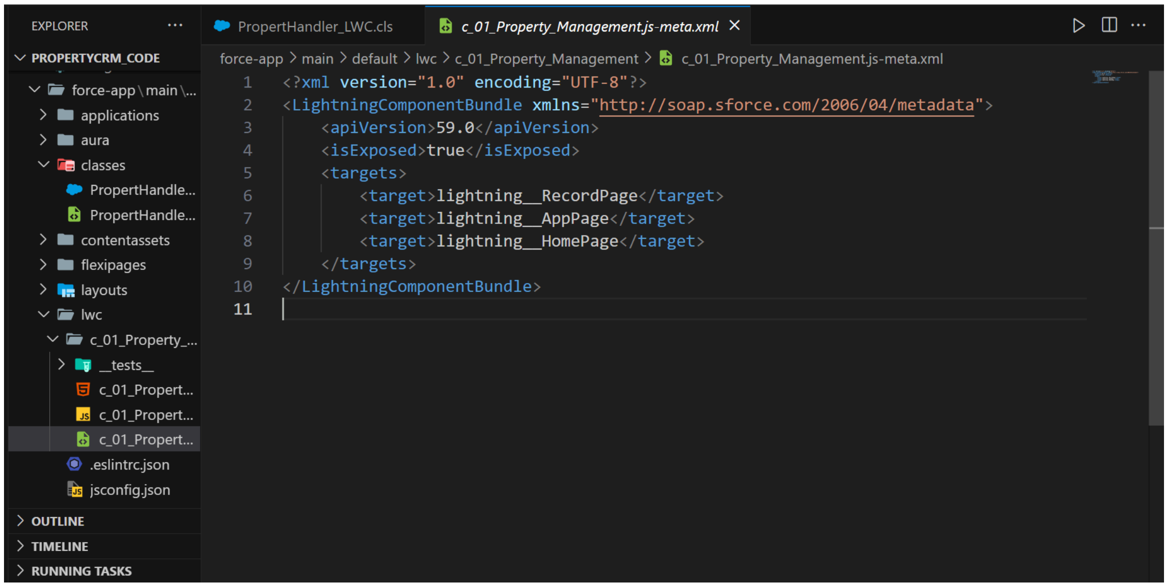
Create a LWC Component



```
public class PropertHandler_LWC {
    @AuraEnabled(cacheable=true)
    public static List<Property_Sheet1__c> getProperty(String type, String verified) {
        return [
            SELECT Id, Location__c, Property_Name__c, Type__c, Verified__c
            FROM Property_Sheet1__c
            WHERE Type__c = :type AND Verified__c = :verified
        ];
    }
}
```



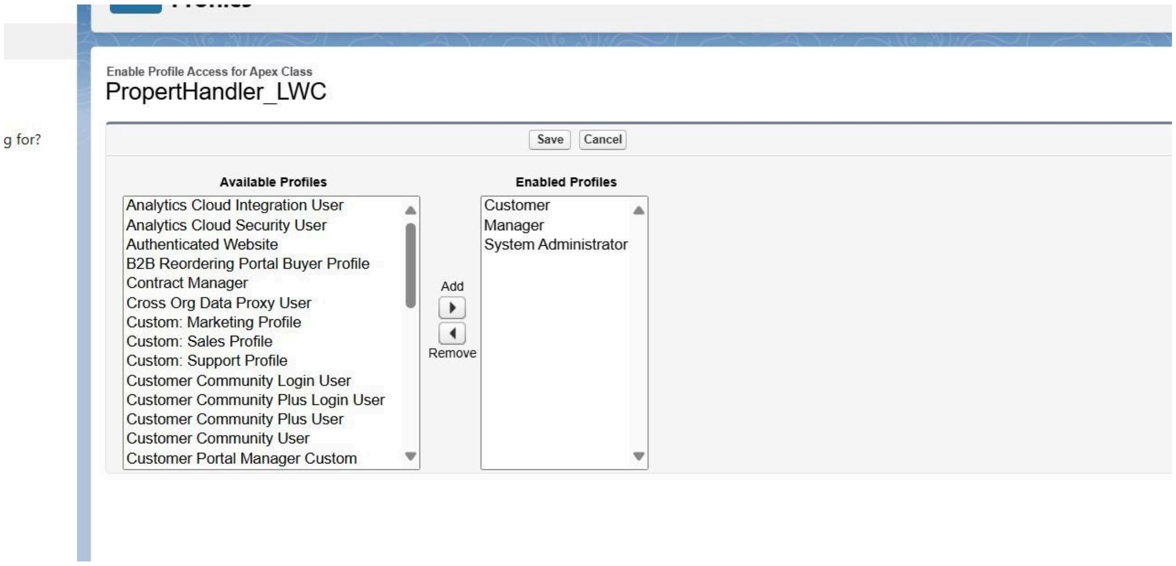
```
<template>
    <lightning-card>
        <div class="slds-box">
            <div class="slds-text-align_left">
                <h1 style="font-size: 20px;"><b>Properties</b></h1>
            </div>
            <div>
                <div class="slds-grid slds-gutters">
                    <div class="slds-col slds-size_5-of-6">
                        <lightning-combobox name="Type" label="Property Type" value={typevar}
                            placeholder="Select Property type" options={propetyoptions}
                            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>

        <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>
```



```xml
<?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>
```

- **Go to Developer Console**
- **Click the gear icon (◦ⓧO) in the top-right corner.**
- **Select Developer Console from the dropdown menu.**
- **Create a New Apex Class.**
- **In the Developer Console, click File in the top left.**
- **Select New and then Apex Class.**
- **Name the Apex Class**
- **In the prompt that appears, name the class PropertyHandler_LWC.**
- **Click OK to create the new class.**
- **Write the Apex Code.**
- **Below is a simple example of an AuraEnabled Apex class that fetches Property records from Salesforce:**

Give Access of Apex Classes to Profiles


Enable Profile Access for Apex Class

PropertHandler_LWC

Available Profiles	Enabled Profiles
Analytics Cloud Integration User	Customer
Analytics Cloud Security User	Manager
Authenticated Website	System Administrator
B2B Reordering Portal Buyer Profile	
Contract Manager	
Cross Org Data Proxy User	
Custom: Marketing Profile	
Custom: Sales Profile	
Custom: Support Profile	
Customer Community Login User	
Customer Community Plus Login User	
Customer Community Plus User	
Customer Community User	
Customer Portal Manager Custom	

- **Search for Apex Classes**
    - **In the Quick Find box on the left sidebar, type Apex Classes and select it.**
- **Locate Your Apex Class**
    - **In the list of Apex classes, find PropertyHandler_LWC.**
- **Click on Security**
    - **Next to PropertyHandler_LWC, click on the Security link. This will take you to the Apex Class Access settings for that specific class.**
- **Modify Class Access for Profiles**
    - **In the Apex Class Access section, you'll see a list of profiles that can be assigned**

      access to this class.
    - **Click the Edit button to modify the access settings.**
- **Add Profiles**
    - **In the available profiles section, search for and select the following profiles:**
        - **Manager**

        - **Customer**

- **You can select the profiles by checking the boxes next to their names.**
- **Save Your Changes**
                - **After adding the desired profiles, click the Save button to apply the changes.**

Conclusion

The development of the Property Management Application in Salesforce has significantly improved the ability to manage properties efficiently. The combination of Apex classes, Lightning components, and custom workflows has streamlined operations and enhanced the user experience. With ongoing maintenance and updates based on user feedback, this application can continue to evolve and meet the needs of its users effectively.