

Project Report

A CRM Application to Handle Client and Property-Related Requirements

Tumuluri Jaya Surya Sasank

jayasuryasasank@gmail.com

Table of Contents

Abstract.....	2
Introduction.....	3
Scope.....	3
Objectives.....	4
Procedure.....	5
Technical Implementation.....	21
Conclusion.....	23
Future Scope.....	23
Output.....	24

Abstract

The "**PropSync Properties**" project integrates Salesforce CRM to create a robust and efficient property management solution for the real estate market. By automating customer interaction workflows, the application captures user details, processes their requests, and provides tailored property recommendations. It categorizes users into approved and non-approved groups, ensuring a personalized user experience. This project exemplifies how a Salesforce-powered CRM system can transform operations, improve engagement, and drive growth in real estate.

Introduction

The real estate sector faces challenges in managing client relationships and meeting customer expectations. With numerous inquiries and properties, it becomes essential to have an automated system that handles customer requirements while providing accurate property matches. PropSync Properties addresses these challenges by integrating Jotform and Salesforce, ensuring seamless interaction between customers and businesses.

This project simplifies client management, property approvals, and data organization, offering real-time property recommendations. The solution also incorporates role-specific access controls to maintain data integrity and secure operations.

Scope

1. Automate the collection of client data.
2. Provide tailored property recommendations based on client preferences.
3. Streamline the approval process for property listings.
4. Offer role-based data access for security and efficiency.
5. Enhance user satisfaction with a clean, interactive interface.

Objectives


The primary goals of this project include:

1. **Automation:** Minimize manual intervention by automating client data collection and property recommendation processes.
2. **Efficiency:** Organize client data into actionable categories for better business insights.
3. **User Experience:** Deliver a personalized, user-friendly platform that caters to customer needs.
4. **Security:** Implement role-based access to ensure only authorized personnel can manage sensitive information.
5. **Scalability:** Design the solution to support future expansion and additional features.

Procedure

Step 1: Data Collection and Form Integration

- **Tool:** Jotform
- **Implementation:**
 - Designed a customer-facing web form for collecting user data such as name, contact details, and property preferences.



PROPSYNC
Streamlining Real Estate with Smart Salesforce Integration

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 *

\$ 0.00

Address

Street Address

Street Address Line 2

City State / Province

Postal / Zip Code


Submit


- o Published the form and integrated it with Salesforce for automatic record creation.


BUILD


SETTINGS


PUBLISH


 **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


 **WORKFLOWS** / Formerly Approvals
Turn your form into a workflow


 **JOTFORM SIGN**
Power your forms with Jotform Sign


 **INTEGRATIONS**
Connect your form to other apps


See All


Search


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

**Square**
Collect Square payments directly through your forms

**Google Sheets**
Instantly populate your spreadsheets with form data

**PayPal Personal**
Collect payments through your online forms

**Google Drive**
Sync file uploads and form submissions to Google Drive

**Twilio**
Send confirmation codes via SMS message

- o Jotform fields were mapped to Salesforce fields to maintain data_integrity.

PROPSYNC

Last edited yesterday.

BUILD

SETTINGS

PUBLISH

Find the record that matches with the selected fields

Object Fields

PROPSYNC

Customer__c

Name - First Name

×

State

Address - State

×

City

Address - City

×

Property Type

Which type of Property are you lookin...

×

Street Address

Address - Street Address

×

Street Address line 2

Address - Street Address 2

×

Name

Name - Last Name

×

postal code

Address - Postal/Zip Code

×

Emial

Email

×

Budget Amount

Budget Amount

×

Phone Number

Phone Number

×

+ Add Field

Create a record

OFF

- **Result:** Automated and accurate collection of customer details.

SALESFORCE

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

All Actions

See Action Logs

+ Add New Action

1

Find existing record

Customer

Step 2: Custom Object Creation

- **Customer Object:** Captures client information such as contact details and preferences.

Custom Object

Customer

Custom Object Definition Edit

SaveSave & NewCancel

Custom Object 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

Example: Account

Plural Label

Customer

Example: Accounts

Starts with vowel sound

☐

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

Object Name

Customer

Example: Account

Description

Fields & Relationships

15 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE
Budget Amount	Budget_Amount__c	Number(18, 0)
City	City__c	Text(255)
Created By	CreatedById	Lookup(User)
Customer	Customer__c	Text(255)
Customer	Name	Text(80)
Email	Email__c	Email
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
Phone Number	Phone_Number__c	Phone
postal code	postal_code__c	Text(255)
Property Type	Property_Type__c	Text(255)
State	State__c	Text(255)
Street Address	Street_Address__c	Text(255)
Street Address line 2	Street_Address_line_2__c	Text(255)
Verified	Verified__c	Checkbox

- **Property Object:** Maintains property-specific details like location, type, and owner.

Edit Custom Object
Property

Custom Object Definition Edit Save Save & New Cancel

Custom Object 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 Example: Account

Plural Label Example: Accounts

Starts with vowel sound ☐

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

Object Name Example: Account

Description

Context-Sensitive Help Setting ☒ Open the standard Salesforce.com Help & Training window
☐ Open a window using a Visualforce page

Content Name

Fields & Relationships			Quick I
8 Items, Sorted by Field Label			
FIELD LABEL	FIELD NAME	DATA TYPE	
Created By	CreatedById	Lookup(User)	
Last Modified By	LastModifiedById	Lookup(User)	
Location	Location__c	Text(255)	
Owner	OwnerId	Lookup(User,Group)	
Property	Name	Text(80)	
Property Name	Property_Name__c	Text(255)	
Type	Type__c	Text(255)	
Verified	Verified__c	Checkbox	

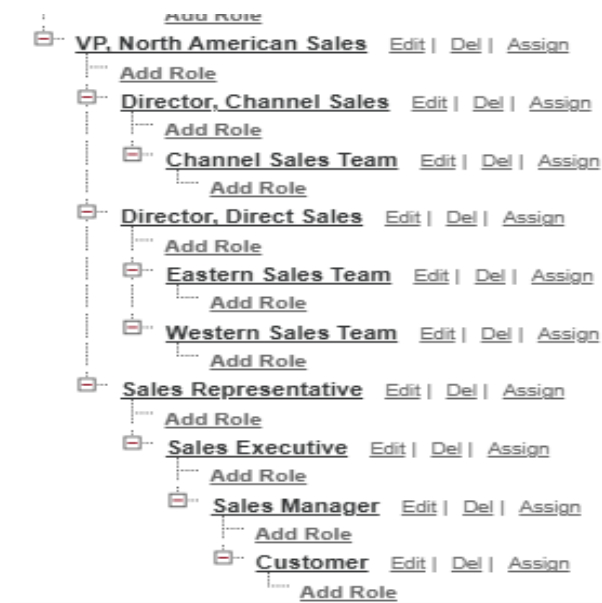
- **Methodology:**

- Utilized the "Create Object from Spreadsheet" feature in Salesforce for efficient object creation.
- Mapped data fields between spreadsheets and Salesforce.

Step 3: Role and Profile Setup

- **Roles Created:**

- Sales Executive: Handles property listings and interacts with potential customers.
- Sales Manager: Oversees sales executives and approves critical operations.
- Customer: End users who search for properties.



- **Profiles Configured:**

- Restricted access to non-relevant objects and ensured each role had permissions tailored to their tasks.

Step 4: Property Approval Workflow

- Designed an approval process to ensure all property listings are verified before being made public.

The screenshot shows the 'Approval Processes' setup page for 'Property: Property Approval'. The page includes a 'Process Definition Detail' section with a table of configuration options. The 'Active' checkbox is checked. The 'Entry Criteria' field contains the expression: `{Property: Location NOT EQUAL TO blank} AND {Property: Verified EQUALS False}`. The 'Record Editability' is set to 'Administrator OR Current Approver'. The 'Approval Assignment Email Template' is set to 'Property Owner: Role: Sales Manager'. The 'Initial Submitters' are 'Property Owner: Role: Sales Manager'. The 'Created By' is 'Java Surya Sasank Tumuluri' and the 'Modified By' is 'Java Surya Sasank Tumuluri'.

Process Definition Detail		Active
Process Name	Property Approval	<input checked="" type="checkbox"/>
Unique Name	Property_Approval	Next Automated Approver Determined By: Manager of Record Submitter
Description		
Entry Criteria	{Property: Location NOT EQUAL TO blank} AND {Property: Verified EQUALS False}	
Record Editability	Administrator OR Current Approver	Allow Submitters to Recall Approval Requests: <input type="checkbox"/>
Approval Assignment Email Template		
Initial Submitters	Property Owner: Role: Sales Manager	
Created By	Java Surya Sasank Tumuluri, 18/11/2024, 11:26 pm	Modified By: Java Surya Sasank Tumuluri, 18/11/2024, 11:31 pm

○ Criteria for approval included:

- Location field must not be empty.
- Property must be marked as "unverified."


The screenshot shows the 'Field Update Edit' form for 'Verified Property'. The 'Name' field is 'Verified Property' and the 'Unique Name' is 'Verified_Property'. The 'Object' is 'Property' and the 'Field to Update' is 'Property: Verified'. The 'Field Data Type' is 'Checkbox'. The 'Re-evaluate Workflow Rules after Field Change' checkbox is unchecked. The 'Specify New Field Value' section shows 'Checkbox Options' with 'True' selected.

Field Update Edit	
Identification	
Name	Verified Property
Unique Name	Verified_Property
Description	
Object	Property
Field to Update	Property: Verified
Field Data Type	Checkbox
Re-evaluate Workflow Rules after Field Change	<input type="checkbox"/>
Specify New Field Value	
Checkbox Options	
<input checked="" type="radio"/> True	
<input type="radio"/> False	

Field Update Edit Save Save & New Cancel

Identification

Name


Unique Name 

Description

Object Property

Field to Update Property: Verified

Field Data Type Checkbox

Re-evaluate Workflow Rules after Field Change ☐ 

Specify New Field Value

Checkbox Options

☐ True

☒ False

Save Save & New Cancel

○ Approval hierarchy:

1.Sales Executive for initial review.

2.Sales Manager for final approval.

Step 5: Application Development


- **Tool:** Salesforce Lightning App Builder
- **App Name:** Property Details


New Lightning App


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 Name 

*Developer Name 

Description 

App Branding

Image 

Primary Color Hex Value

Org Theme Options
☐ Use the app's image and color instead of the org's custom theme

App Launcher Preview

Next


- **Features:**

- Consolidates customer and property objects.
- Provides a user-friendly interface for managing records.

Step 6: Automation Through Flows

- **Record Trigger Flow:**

- Automatically submits property records for approval upon creation.
- Simplifies the approval process and reduces delays.

 Flow

Property Approval

Related

Details

Information

Flow Label

Property Approval

Description

Associated Record

Created By

Last Modified

Category

API Name

Property_Approval

Flow Type

Record-Triggered After Save Flow

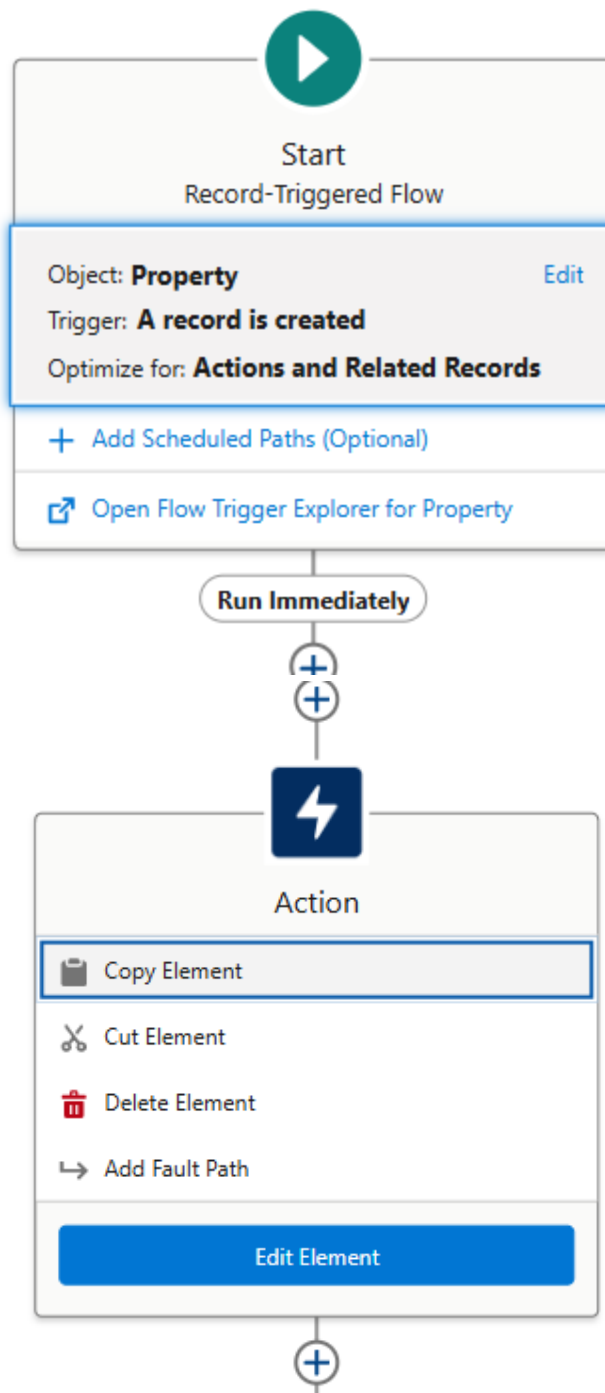
Created Date

18/11/2024, 11:36 pm

Last Modified Date

18/11/2024, 11:36 pm

Subcategory



Step 7: Lightning Web Component (LWC)

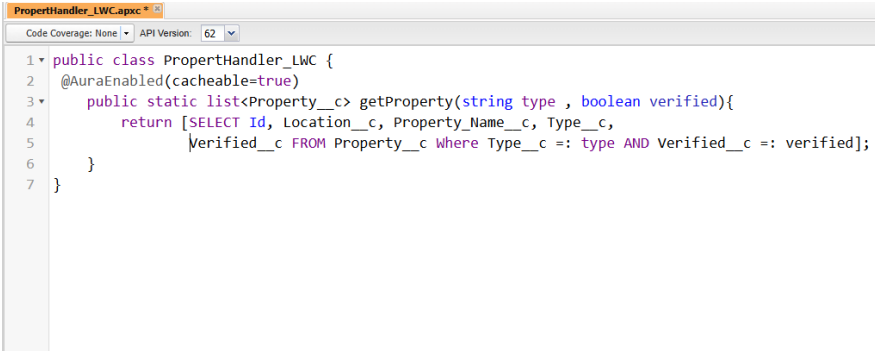
- **Purpose:**

- Allows verified customers to access approved properties.
- Restricts non-verified customers to view only non-approved properties.

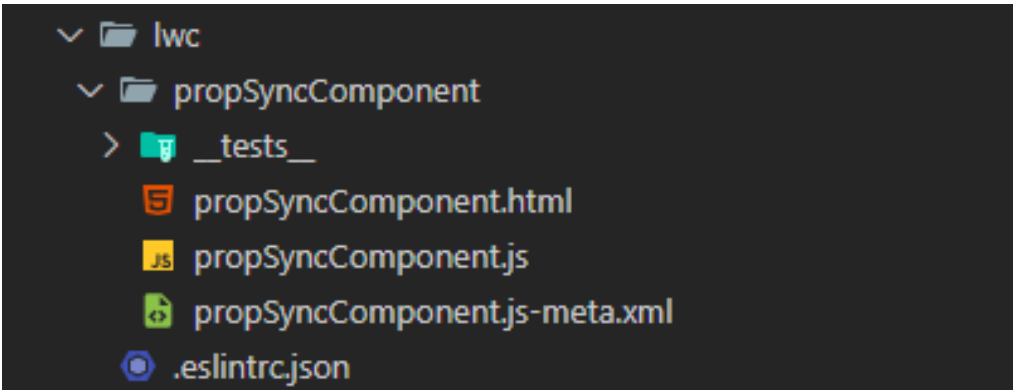
- **Implementation:**

- Created an Apex class for querying verified properties.

Developed an interactive LWC component with property filters.

○ The screenshot shows the Salesforce IDE editor for a class named 'PropertHandler_LWC.apex'. The code is as follows:

```
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,  
5                 Verified__c FROM Property__c Where Type__c =: type AND Verified__c =: verified];  
6     }  
7 }
```

○ The screenshot shows a file explorer view of a Lightning Web Component project. The directory structure is as follows:

- lwc
 - propSyncComponent
 - __tests__
 - propSyncComponent.html
 - propSyncComponent.js
 - propSyncComponent.js-meta.xml
 - .eslintrc.json


```

<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>
    <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>
    </div>
  </lightning-card>
</template>

```

○

```

import { LightningElement, api, track, wire } from 'lwc';
import getProperty from '@salesforce/apex/PropertyHandler_LWC.getProperty';
import { getRecord } from 'lightning/uiRecordApi';
import USER_ID from '@salesforce/user/Id';

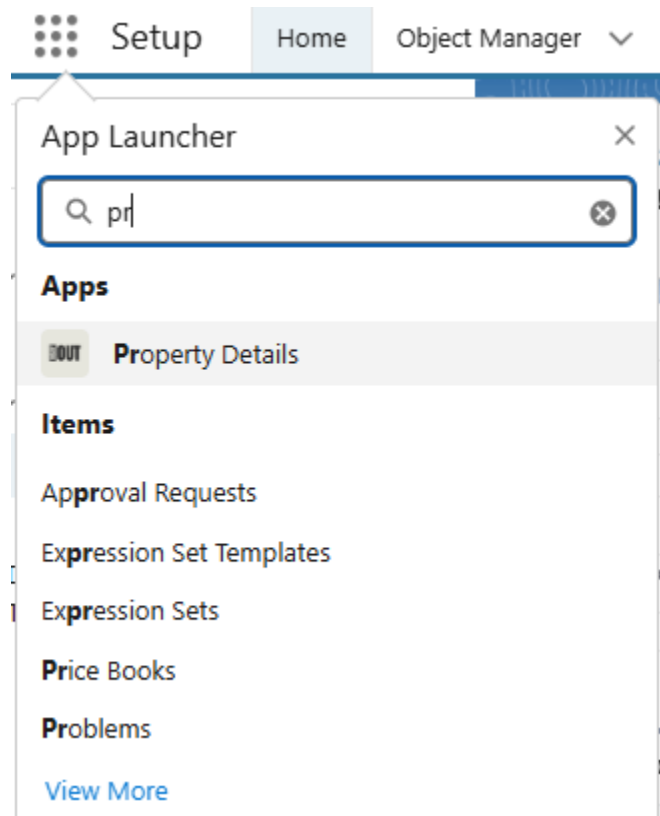
export default class PropSyncComponent extends LightningElement {
    @api recordId;
    userId = USER_ID;
    verifiedvar;
    typevar;
    isfalse = true;
    istrue = false;
    @track propertylist = [];
    columns = [
        { label: 'Property Name', fieldName: 'Property_Name__c' },
        { label: '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: '$recordId', 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)
            })
    }
}

```

O

- **Deployment:**

- Added the component to the "Search Your Property" app page for easy access.




Step 8: Deployment and Security

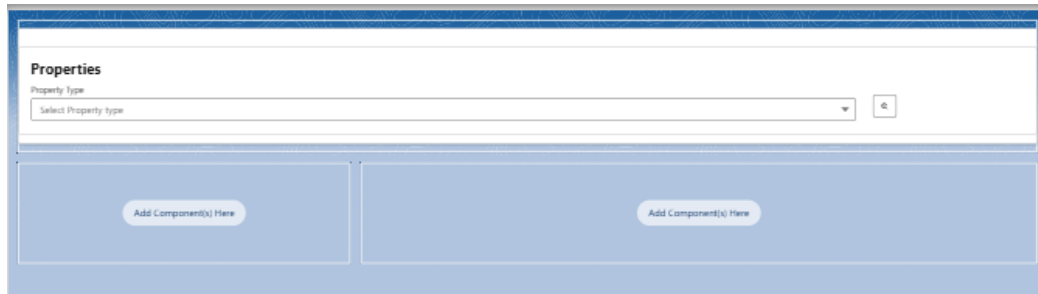
- **Deployment:**

- Deployed the app, components, and workflows into the production environment.
- Activated pages and profiles for end-user interaction.

▼ Custom (1)

 propSyncComponent

○



- **Security:**

- Configured Apex class access for relevant roles.
- Ensured that role-based permissions were strictly enforced.

Technical Implementation

Technologies Used

- **Salesforce CRM:** For data organization, workflows, and app development.
- **Jotform:** For creating and integrating customer-facing forms.
- **Lightning Web Components (LWC):** For creating dynamic user interfaces.
- **Apex:** For backend logic to support LWC operations.

Integration Details

- **Jotform Integration:**
 - Automatically pushes data to Salesforce objects upon form submission.
 - Reduces manual data entry errors.
- **Approval Process:**
 - Configured an automated flow for verifying property details.
 - Ensures compliance with business requirements.

User Experience

- Implemented a clean, intuitive interface for property search.
- Enhanced search options with filters for property type and verification status.

Results

1. **Automation:** Customer data collection and property approvals are fully automated, minimizing human intervention.
2. **Efficiency:** Streamlined workflows reduced turnaround times for property listings.
3. **User Satisfaction:** A tailored property search experience increased engagement.
4. **Business Growth:** Simplified operations allow scalability and attract more clients.

Challenges

1. Complex role and permission configurations for various business requirements.
2. Ensuring the security of customer and property data during integrations.
3. Optimizing workflows to handle large volumes of records efficiently.

Conclusion

The "PropSync Properties" project successfully addresses the challenges in real estate management by integrating Salesforce with automated workflows. It enhances the customer experience, optimizes operations, and supports scalability. The solution is a benchmark for leveraging CRM tools in the real estate sector.

Future Scope

1. AI-Driven Recommendations:

- a. Use machine learning to analyze customer preferences and suggest properties.

2. Advanced Reporting:

- a. Add dashboards for insights into sales performance and customer behavior.

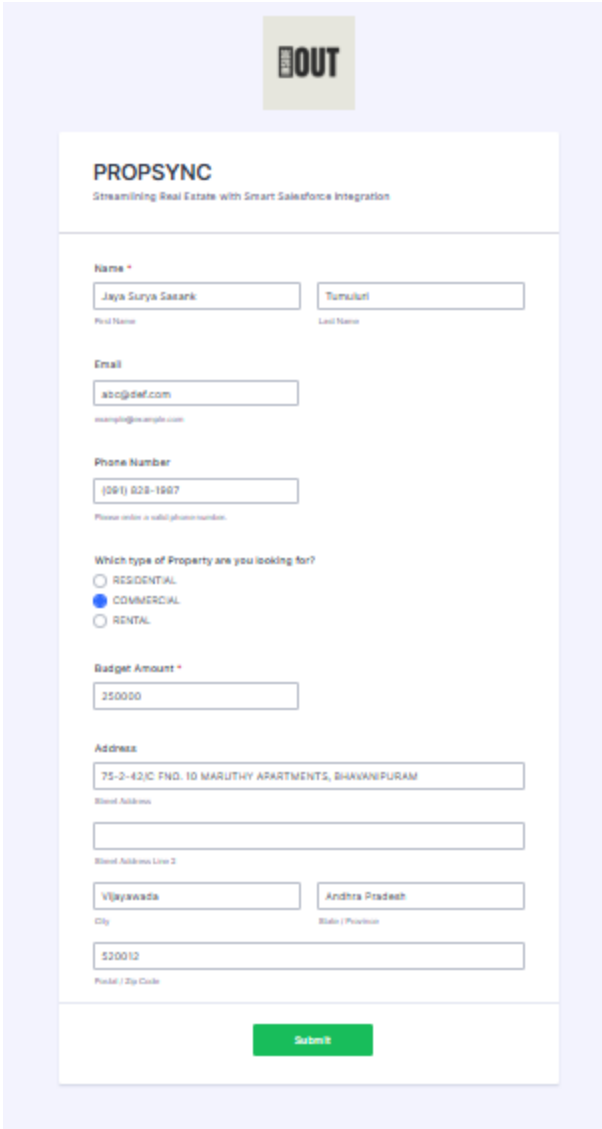
3. Mobile Integration:

- a. Develop a mobile-friendly version for better accessibility.

Output

1. Customer Form Link: [Jotform Link](#)

2. Approval Process Screenshots



The screenshot shows a web form titled "PROPSYNC" with the subtitle "Streamlining Real Estate with Smart Salesforce Integration". At the top left of the form is a logo that says "LEAD OUT". The form contains the following fields and sections:

- Name ***: Two input fields. The first is labeled "First Name" and contains "Jaya Surya Sasank". The second is labeled "Last Name" and contains "Tumuluri".
- Email**: One input field containing "abc@def.com". Below it is a small text "example@gmail.com".
- Phone Number**: One input field containing "(081) 828-1987". Below it is a small text "Please enter a valid phone number".
- Which type of Property are you looking for?**: Three radio buttons. "RESIDENTIAL" is unselected, "COMMERCIAL" is selected (indicated by a blue dot), and "RENTAL" is unselected.
- Budget Amount ***: One input field containing "250000".
- Address**: One input field containing "75-2-42/C FNO. 10 MARUTHY APARTMENTS, SHAKUNIPURAM". Below it is a small text "Street Address".
- Street Address Line 2**: One empty input field.
- City**: One input field containing "Vijayawada". Below it is a small text "City".
- State / Province**: One input field containing "Andhra Pradesh". Below it is a small text "State / Province".
- Postal / Zip Code**: One input field containing "520012". Below it is a small text "Postal / Zip Code".
- Submit**: A green button at the bottom center of the form.



Thank You!

Your submission has been received.

Now create your own Jotform - It's free!

Create your own Jotform

DOT

Search...

EPT: 1.55 s 3799.4 KB

Property Details Search your Property Property Customer

Customer

Recently Viewed

New Change Owner Import Assign Label

1 Item • Updated a few seconds ago

Customer Customer Phone Nu... Emial State Property ... Budget A... Street Ad... Street Ad... City

1 Tumuluri Jaya Surya... (091) 828-... abc@def... Andhra Pr... COMMER... 2,50,000 75-2-42/C... Vijayawada

DOT

Search...

EPT: 0.9 s 4613.48 KB

Property Details Search your Property Property Customer

Customer

All Records

New Change Owner Import Printable View Assign Label

4 Items • Sorted by Customer • Filtered by All customer • Updated a few seconds ago

Customer Customer Phone Num... Emial State Property T... Budget... Street Address Street Ad... City postal...

1 a00dL00000UWn... Rakesh 788797.0 rakesh@gmail.com Telangana Residential 40,00,000 gb road street no 45 Hyderabad 555001

2 a00dL00000UWnse prakash 55448855 p@gmail.com Maharashtra Commercial 80,00,000 gachibowli indira road mumbai 66000...

3 a00dL00000UWnaf Prajwal 454545.0 prajwal@gmail.com Maharashtra Rental 25,000 kamdli kathora Amravati 444805

4 Tumuluri Jaya Surya Sasa... (091) 828-1987 abc@def.com Andhra Prade... COMMERCIAL 2,50,000 75-2-42/C FNO. 10 ... Vijayawa...

OUT

Q Search...

EPT: 0.52 s 5971.41 KB

+

?

12

Property Details

Search your Property

Property

Customer

Property

All Records

New

Change Owner

Import

Printable View

Assign Label

Q Search this list...

⚙

📄

🔄

✎

🗑

🔍

3 Items • Sorted by Property • Filtered by All property • Updated a few seconds ago

	<input type="checkbox"/> Property ↑	Property Name	Type	Location	Verified	
1	<input type="checkbox"/> a01dL00000dKJOM	Lotus Apartments	Residential	hyderabad	<input type="checkbox"/>	⌵
2	<input type="checkbox"/> a01dL00000dKJON	50000 sq.ft plot	Commercial	Amravati	<input type="checkbox"/>	⌵
3	<input type="checkbox"/> a01dL00000dKJOO	3 Bhk flat at stanza	rental	Jubilee hill Hydeeabad	<input type="checkbox"/>	⌵

Property Details

Search your Property

Property

Customer

Search your Property

Properties

Property Type

Select Property type

Commercial

Residential

rental

🔍

Property Details

Search your Property

Property

Customer

Search your Property

Properties

Property Type

Commercial

🔍

<input type="checkbox"/> Property Name	Property Type	Property Location	Property link
<input type="checkbox"/> 50000 sq.ft plot	Commercial	Amravati	

Property Details

Search your Property

Property

Customer

Search your Property

Properties

Property Type

Residential

🔍

<input type="checkbox"/> Property Name	Property Type	Property Location	Property link
<input type="checkbox"/> Lotus Apartments	Residential	hyderabad	

OUT

Q Search...

EPT: 0.47 s 6091.32 KB

+

?

12

Property Details

Search your Property

Property

Customer

Search your Property

Properties

Property Type

rental

🔍

<input type="checkbox"/> Property Name	Property Type	Property Location	Property link
<input type="checkbox"/> 3 Bhk flat at stanza	rental	Jubilee hill Hydeeabad	

3. Apex Class 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];  
    }  
}
```

4. LWC Component Code

propSyncComponent.js:

```
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 PropSyncComponent 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" }
    ]
    propetyoptions = [
        { 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)  
})  
}  
}
```

propSyncComponent.html:

```
<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">

<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">

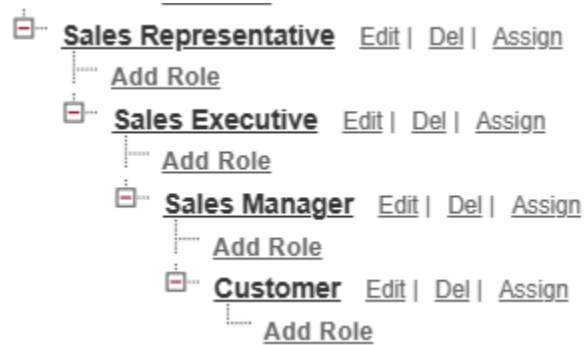
<lightning-datatable key-field="id" data={propertylist}
columns={columns}></lightning-datatable>

```
</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>
```

propSyncComponent.js-meta.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle
xmlns="http://soap.sforce.com/2006/04/metadata">
  <apiVersion>62.0</apiVersion>
  <isExposed>true</isExposed>
  <targets>
    <target>lightning__RecordPage</target>
    <target>lightning__AppPage</target>
    <target>lightning__HomePage</target>
  </targets>
</LightningComponentBundle>
```


5. Role Hierarchy and Profiles Configuration



Profile Edit
Customer [Help for this Page](#)

Set the permissions and page layouts for this profile.

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

Name

User License **Salesforce Platform** Custom Profile ☒

Description

Custom App Settings [Required Information](#)

	Visible	Default		Visible	Default
Analytics Studio (standard__Insights)	<input type="checkbox"/>	<input type="radio"/>	Property Details (Property_Details)	<input checked="" type="checkbox"/>	<input checked="" type="radio"/>
App Launcher (standard__AppLauncher)	<input type="checkbox"/>	<input type="radio"/>	WDC (standard__Work)	<input type="checkbox"/>	<input type="radio"/>
Platform (standard__Platform)	<input type="checkbox"/>	<input type="radio"/>			

Profile Edit
Manager [Help for this Page](#)

Set the permissions and page layouts for this profile.

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

Name

User License **Salesforce Platform** Custom Profile ☒

Description

Custom App Settings [Required Information](#)

	Visible	Default		Visible	Default
Analytics Studio (standard__Insights)	<input type="checkbox"/>	<input type="radio"/>	Property Details (Property_Details)	<input checked="" type="checkbox"/>	<input type="radio"/>
App Launcher (standard__AppLauncher)	<input type="checkbox"/>	<input type="radio"/>	WDC (standard__Work)	<input type="checkbox"/>	<input type="radio"/>
Platform (standard__Platform)	<input checked="" type="checkbox"/>	<input checked="" type="radio"/>			