# **A CRM Application to Handle the Clients and their Property-Related Requirements**

## **Project Overview**

**Project Name:** A CRM Application to Handle the Clients and their Property-Related Requirements  
**By:** SHAIK ARSHAD UZZAMA  
**Email ID:** uzzama\_shaik@srmap.edu.in

### **Project Abstract:**

Dreams World Properties integrates Salesforce to streamline customer interactions and property management processes. Through a seamless integration between Jotform, Salesforce, and custom Lightning components, the CRM automates customer data entry, categorizes users (approved or non-approved), and provides personalized property recommendations. By incorporating Salesforce’s powerful automation, reporting, and role-based security features, the CRM optimizes operations, enhances customer satisfaction, and facilitates business growth in the real estate sector.

## **Table of Contents**

1. **Introduction**
2. **Project Objectives**
3. **Detailed Implementation**
   1. Integration with Jotform
   2. Creating Salesforce Objects
   3. Role Management
   4. Property Details App Creation
   5. Profiles Configuration
   6. Apex Class and LWC Integration
   7. Apex Class Security for Profiles
4. **Conclusion**

## **Introduction**

The CRM system is designed to simplify the management of client relationships and property-related requirements through Salesforce. By automating workflows, enabling role-based access, and incorporating real-time property searches based on customer preferences and approval status, the CRM enhances both operational efficiency and the customer experience. The system integrates Salesforce with tools like Jotform, custom Apex classes, and Lightning Web Components (LWC) to ensure real-time data synchronization and personalized recommendations for users.

## **Project Objectives**

1. **Improve Customer Engagement**:  
   By providing real-time property recommendations based on a client’s verified status, we ensure a tailored, engaging user experience that enhances customer satisfaction and loyalty.
2. **Automated Data Collection**:  
   Integrating Jotform for customer data collection automates the process of entering customer information into Salesforce, reducing human error and saving time.
3. **Role-Based Access Control**:  
   Different user profiles (Sales Executive, Manager, Customer) will have specific permissions based on their roles, ensuring security and data integrity. The system will offer permissions according to the verification status (approved or non-approved).
4. **Enhanced Search Functionality**:  
   Users will be able to search properties by type (commercial, residential, rental) and based on their approval status. A powerful LWC with integrated Apex classes will help provide these search functionalities.

## **Detailed Implementation**

### **1. Integration with Jotform**

**Goal:**Automate the process of capturing customer details and integrating this data into Salesforce, minimizing manual data entry.

**Steps:**

1. **Create a Jotform Form:**
   * Log in to your Jotform account.
   * Click on **Create Form** and choose **Start From Scratch**.
   * Add the necessary fields (Name, Phone Number, Email, Address, Preferred Property Type, Budget, etc.).
   * After designing the form, click **Publish** and obtain the form URL for distribution.
2. **Integrating Jotform with Salesforce:**
   * Navigate to **Jotform Integration** within the Jotform dashboard.
   * Search for **Salesforce** and connect your Jotform account to Salesforce using an API key (ensure you have the right Salesforce credentials).
   * **Map Form Fields to Salesforce**: Map each field from the Jotform form to the Salesforce **Customer Object** fields.
     + Example:
       - Jotform Field "Name" → Salesforce Field "Customer Name"
       - Jotform Field "Email" → Salesforce Field "Email"
       - Jotform Field "Preferred Property Type" → Salesforce Field "Property Type"
3. **Salesforce Setup**:
   * Create a custom **Customer** object in Salesforce.
     + Navigate to **Setup** → **Object Manager** → **Create** → **Custom Object**.
     + Name it "Customer," and add necessary fields like Name, Email, Phone, Property Preferences, Status (Approved/Non-approved), etc.
   * Once integrated, customer records will be automatically created in Salesforce every time a Jotform submission is received.
4. **Automation via Workflow**:  
   Set up a workflow in Salesforce that will automatically send an email or trigger a notification when a new customer submits their data through Jotform.

### **2. Creating Salesforce Objects**

To manage and store property and customer details effectively, we will create custom Salesforce objects for both.

**Steps for Creating Salesforce Objects:**

1. **Create a Customer Object**:
   * Navigate to **Setup** → **Object Manager** → **Create Custom Object**.
   * Name the object “Customer” and define relevant fields:
     + **Name** (Text Field)
     + **Email** (Email Field)
     + **Phone Number** (Phone Field)
     + **Preferred Property Type** (Picklist: Residential, Commercial, Rental)
     + **Approval Status** (Picklist: Approved, Non-approved)
     + **Property Preferences** (Long Text Area)
     + **Property** (Relationship field to Property object)
2. **Create a Property Object**:
   * Navigate to **Setup** → **Object Manager** → **Create Custom Object**.
   * Name the object “Property” and define fields such as:
     + **Property Name** (Text Field)
     + **Type of Property** (Picklist: Residential, Commercial, Rental)
     + **Price** (Currency Field)
     + **Location** (Text Field)
     + **Availability Status** (Picklist: Available, Sold, Under Contract)
     + **Assigned Sales Representative** (Lookup to User)
3. **Relationships**:
   * Establish relationships between the **Customer** and **Property** objects by creating a **Lookup Relationship** field in the **Customer** object, pointing to the **Property** object. This allows customers to be associated with specific properties.

### **3. Role Management**

To ensure proper access control, Salesforce allows the definition of user roles that control data visibility. Here's how to configure them:

1. **Sales Executive Role**:
   * Navigate to **Setup** → **Roles**.
   * Click **Add Role** and name it **Sales Executive**. This role will report to the **Sales Manager**.
   * The **Sales Executive** role has permissions to create, read, and edit customer records but limited access to property management.
2. **Sales Manager Role**:
   * Create the **Sales Manager** role, reporting to **Sales Executive**.
   * **Sales Manager** can manage both **Customer** and **Property** objects, allowing full control of customer and property records.
3. **Customer Role**:
   * Create a **Customer** role that reports to the **Sales Manager**.
   * This role will only have **Read** and **View All** permissions for the **Property** object, restricting them from modifying or managing the data.

### **4. Property Details App Creation**

This application allows users to access and manage customer and property data.

**Steps for App Creation**:

1. **Navigate to Setup** → **App Manager**.
2. Click **New Lightning App** and name it **Property Details**.
3. Add the **Customer** and **Property** objects to the app.
4. Define which tabs and components should be visible to different roles (Sales Manager, Sales Executive, and Customer).

### **5. Profiles Configuration**

**Profiles** define what users can see and do in Salesforce. Let’s define two main profiles:

1. **Customer Profile**:
   * Clone the **Salesforce Platform User** profile.
   * Uncheck access for all objects except **Property**. This allows customers to only view properties.
   * Set field-level security to restrict editing of personal or sensitive information.
2. **Manager Profile**:
   * Clone the **Salesforce Platform User** profile.
   * Assign **Modify All** permissions to both **Customer** and **Property** objects. This allows managers to create, edit, and delete customer profiles and property records.

### **6. Apex Class and LWC Integration**

#### **Apex Class: PropertyHandler\_LWC**

1. **Goal**: Create a class to handle property search queries, filtering based on user’s approval status and property preferences.  
   **Steps**:
   * Navigate to **Developer Console** → **File** → **New** → **Apex Class**.

Create a class called

PropertyHandler\_LWC:

public with sharing class PropertyHandler\_LWC {

@AuraEnabled(cacheable=true)

public static List<Property\_\_c> getProperties(String propertyType, Boolean isApproved) {

List<Property\_\_c> properties = [SELECT Name, Type\_of\_Property\_\_c, Price\_\_c FROM Property\_\_c WHERE Type\_of\_Property\_\_c = :propertyType];

if (!isApproved) {

properties = properties.subList(0, Math.min(10, properties.size()));

}

return properties;

}

}

1. **LWC**:
   * Create a new LWC to allow users to search properties.
   * Connect this component to the PropertyHandler\_LWC Apex class.
2. **Steps**:
   * Use the **lightning-input** component for property type selection.
   * Use **lightning-button** to trigger the search.
   * Display the search results dynamically.

### **7. Apex Class Security for Profiles**

1. **Apex Class Access**:
   * Navigate to **Setup** → **Profiles**.
   * For each relevant profile (e.g., Sales Manager, Customer), assign access to the PropertyHandler\_LWC class.
   * This will enable specific users to utilize the LWC and perform property searches.

## 

## **Conclusion**

The CRM Application effectively combines the power of Salesforce with the ease of automated data collection via Jotform and the flexibility of custom Lightning Web Components and Apex classes. It optimizes operations by automating workflows, providing role-based access to data, and enabling a personalized customer experience. This approach not only enhances operational efficiency but also contributes to a more dynamic, engaging customer experience in the competitive real estate market.