LEASE MANAGEMENT

NAME :K P L PRAVALLIKA
EMAIL ID:pravallikavarma05@gmail.com
COLLEGE:CVR COLLEGE OF ENGINEERING

Project Overview

A **Lease Management System** is developed to automate and manage the processes related to leasing real estate properties, equipment, or other assets. The main objective is to streamline lease operations including tenant management, rent payments, lease lifecycle tracking, and automated notifications using Salesforce as the backend CRM.

Phase 1: Requirement Analysis & Planning

Objectives:

- Understand and define lease-related processes
- Identify user roles and their permissions
- Determine key modules: Property, Tenant, Lease, and Payments
- Plan email communications and approval workflows

Key Requirements:

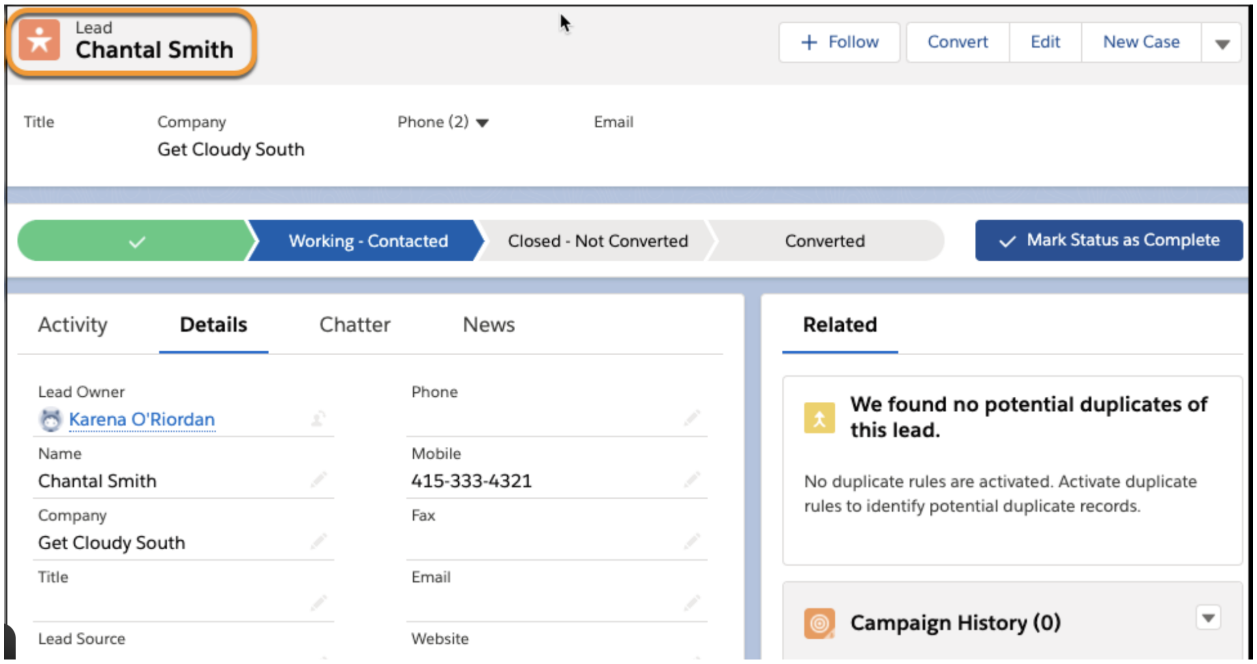
- CRUD operations on lease data
- Relationship between tenants and properties
- Monthly rent payment tracking
- Email alerts and approvals
- Validation rules
- Apex triggers for business logic
- Scheduled email reminders

Phase 2: Backend Setup in Salesforce

Custom Object Creation

1. **Property Object**
    - Label: `Property`
    - Plural Label: `Property`
    - Record Name: `Property Name` (Text)
    - Settings: Allow Reports, Track Field History, Activities, Search
2. **Tenant Object**
    - Label: `Tenant`
    - Plural Label: `Tenants`
    - Record Name: `Tenant Name` (Text)

3. **Payment Object**
    - Label: `Payment for tenant`
    - Plural Label: `Payments`
    - Record Name: `Payment Name` (Text)
4. **Lease Object**
    - Label: `Lease`
    - Plural Label: `Lease`
    - Record Name: `Lease Name` (Text)



Title
Company: Get Cloudy South
Phone (2)
Email

Lead Owner: Karena O'Riordan
Phone
Name: Chantal Smith
Mobile: 415-333-4321
Company: Get Cloudy South
Fax
Title
Email
Lead Source
Website

We found no potential duplicates of this lead.

No duplicate rules are activated. Activate duplicate rules to identify potential duplicate records.

Phase 3: UI/UX Development & Customization

Tabs Creation

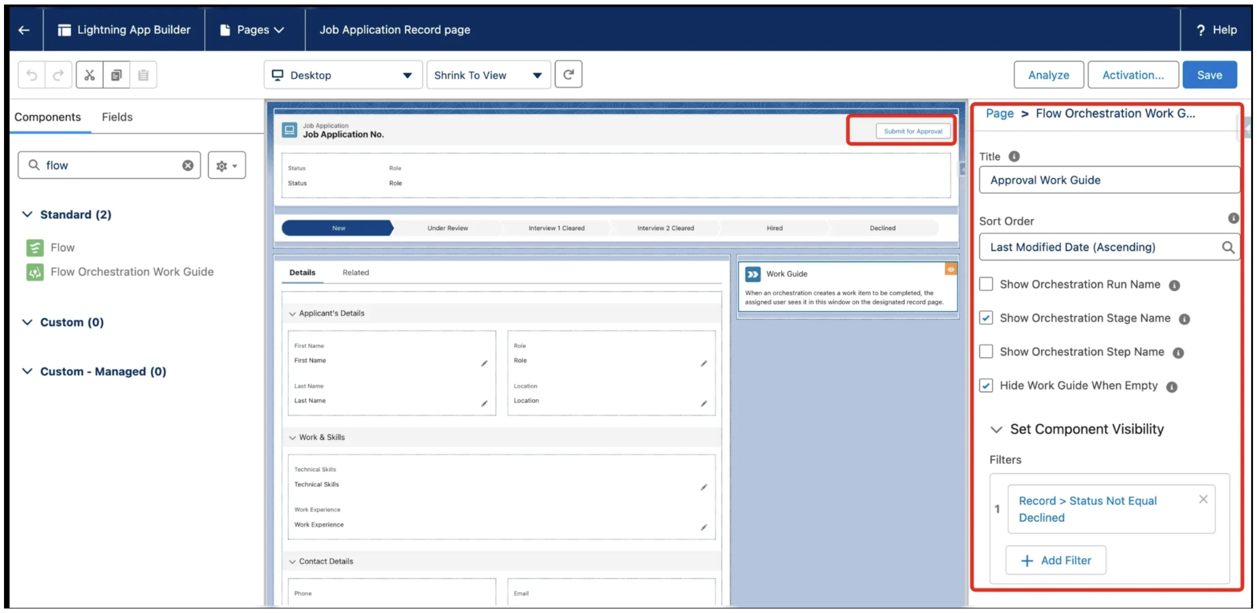
For each custom object:

- Go to: Setup → Tabs → New Custom Object Tab
- Select Object: e.g., Property
- Choose Tab Style
- Add to profiles (default)
- Keep "Append to users' personal customizations" checked

Repeat for: **Tenant**, **Lease**, **Payment for tenant**

Lightning App Creation

1. Go to **App Manager** → New Lightning App
2. App Name: `Lease Management`
3. Navigation Style: Standard
4. Add Navigation Items: Property, Tenants, Lease, Payments
5. Assign to: `System Administrator`
6. Save & Finish



Title: Approval Work Guide

Sort Order: Last Modified Date (Ascending)

- [ ] Show Orchestration Run Name
- [✓] Show Orchestration Stage Name
- [ ] Show Orchestration Step Name
- [✓] Hide Work Guide When Empty

Set Component Visibility

Filters

1 Record > Status Not Equal Declined

Field Creation

➤ Property Object:

- `Name` (Text, Required)
- `Address` (Long Text)
- `Type` (Picklist: 1BHK, 2BHK, 3BHK)
- `sfqt` (Text)

➤ Tenant Object:

- `Email` (Email, Required)
- `Phone` (Phone)
- `Status` (Picklist: Stay, Leaving)

➤ **Lease Object:**

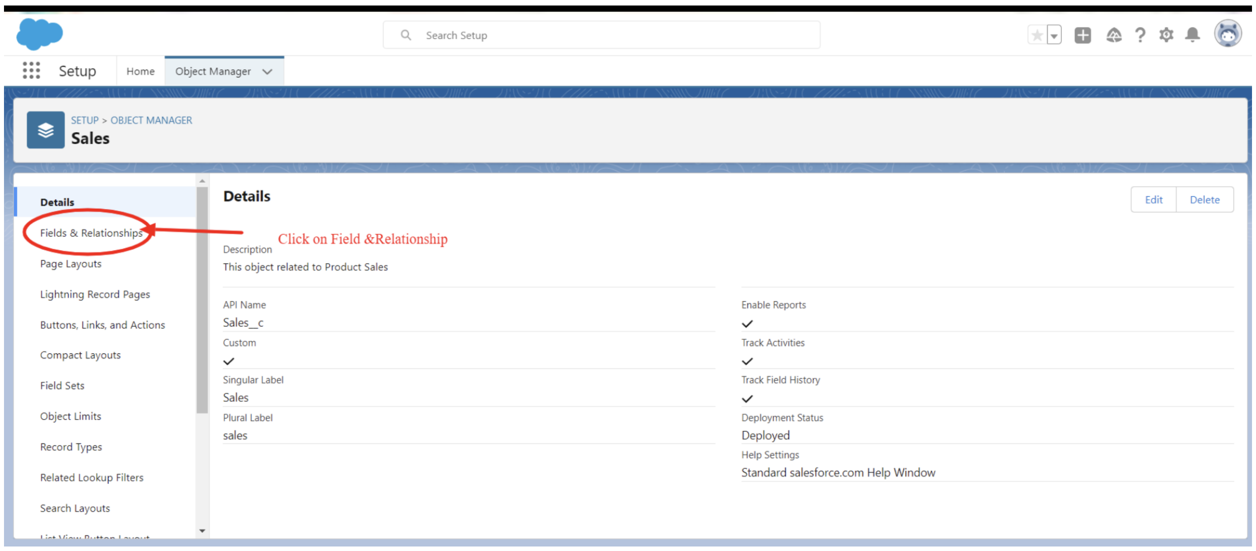
- `Start Date` (Date)
- `End Date` (Date)
- **Lookup to Property** (`Property`)

➤ **Payment for Tenant Object:**

- `Payment Date` (Date)
- `Amount` (Number)
- `Check for Payment` (Picklist: Paid, Not Paid)
- **Lookup to Tenant**

Relationships

- **Lease** → Lookup to `Property`
- **Payment** → Lookup to `Tenant`
- **Property** → Master-Detail on `Payment`



Details

Description
This object related to Product Sales

API Name
Sales__c

Custom
✓

Singular Label
Sales

Plural Label
sales

Enable Reports
✓

Track Activities
✓

Track Field History
✓

Deployment Status
Deployed

Help Settings
Standard salesforce.com Help Window

Phase 4: Data Migration, Testing & Security

Validation Rules

On **Lease Object**:

- **Rule Name**: `lease_end_date`

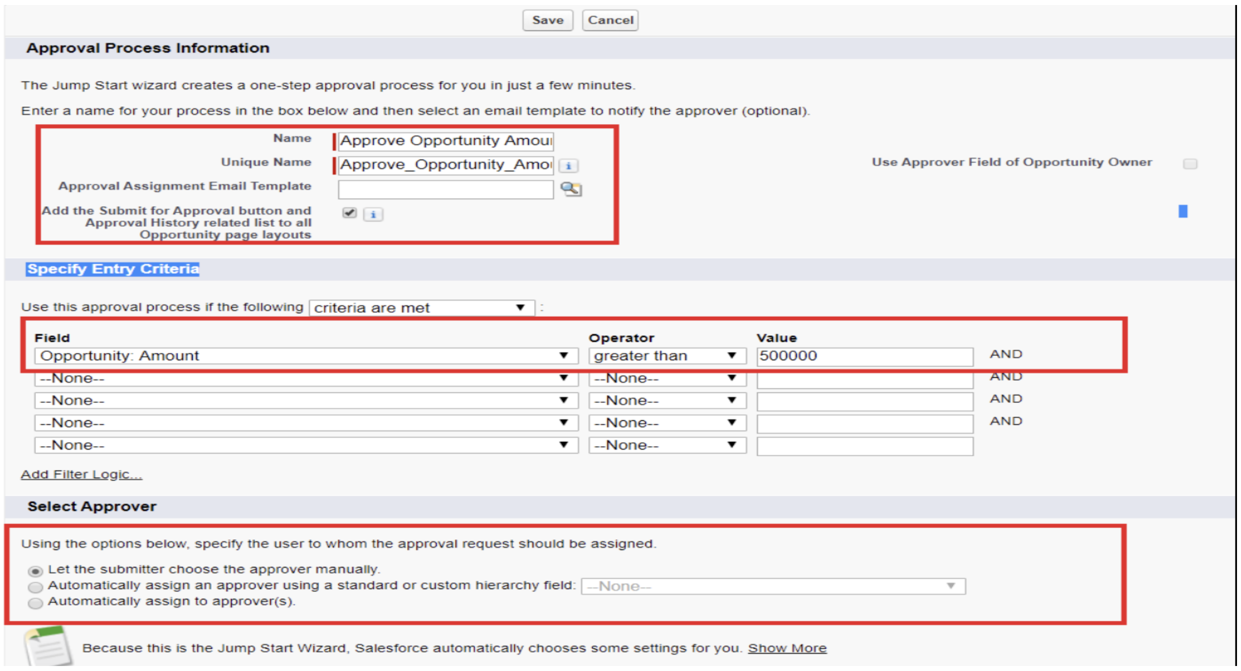
- **Formula**:
  `End_Date__c < Start_Date__c`
- **Error Message**: `"Your End date must be greater than start date"`

Email Templates

Template Name	Subject	Used In
Tenant Leaving	Request for approve the leave	Initial approval
Leave Approved	Leave Approved	Final Approval
Leave Rejected	Leave Rejected	Final Rejection
Tenant Email	Monthly Rent Payment Reminder	Scheduler
Tenant Payment	Confirmation of Successful Monthly Payment	Flow Action

Approval Process – Check for Vacant

- **Object**: Tenant
- **Field Criteria**: Status ≠ Leaving
- **Approver**: Admin
- **Submitter**: Property Owner
- **Actions**:
    - Initial: Send email using `Tenant Leaving`
    - Approved: Send email using `Leave Approved`
    - Rejected: Send email using `Leave Rejected`



Approval Process Information

The Jump Start wizard creates a one-step approval process for you in just a few minutes.

Enter a name for your process in the box below and then select an email template to notify the approver (optional).

Name	Approve Opportunity Amoui
Unique Name	Approve_Opportunity_Amo
Approval Assignment Email Template	
Add the Submit for Approval button and Approval History related list to all Opportunity page layouts	✓
Use Approver Field of Opportunity Owner	☐

Specify Entry Criteria

Use this approval process if the following criteria are met :

Field	Operator	Value	
Opportunity: Amount	greater than	500000	AND
--None--	--None--		AND
--None--	--None--		AND
--None--	--None--		AND
--None--	--None--		

Add Filter Logic...

Select Approver

Using the options below, specify the user to whom the approval request should be assigned.

- [✓] Let the submitter choose the approver manually.
- Automatically assign an approver using a standard or custom hierarchy field: --None--
- Automatically assign to approver(s).

Because this is the Jump Start Wizard, Salesforce automatically chooses some settings for you. Show More

Trigger and Handler

Apex Trigger: `test`

```
trigger test on Tenant__c (before insert) {
    if(trigger.isInsert && trigger.isBefore){
        testHandler.preventInsert(trigger.new);
    }
}
```

Apex Class: `testHandler`

```
public class testHandler {
   public static void preventInsert(List<Tenant__c> newlist) {
        Set<Id> existingPropertyIds = new Set<Id>();
        for (Tenant__c existingTenant : [SELECT Id, Property__c FROM
Tenant__c WHERE Property__c != null]) {
            existingPropertyIds.add(existingTenant.Property__c);
        }

        for (Tenant__c newTenant : newlist) {
            if (newTenant.Property__c != null &&
existingPropertyIds.contains(newTenant.Property__c)) {
                newTenant.addError('A tenant can have only one
property');
            }
        }
    }
}
```

Phase 5: Deployment, Documentation & Maintenance

Scheduled Monthly Email Reminder

Apex Class: `MonthlyEmailScheduler`

```
global class MonthlyEmailScheduler implements Schedulable {
    global void execute(SchedulableContext sc) {
        Integer currentDay = Date.today().day();
        if (currentDay == 1) {
            sendMonthlyEmails();
        }
    }

    public static void sendMonthlyEmails() {
        List<Tenant__c> tenants = [SELECT Id, Email__c FROM
Tenant__c];
        for (Tenant__c tenant : tenants) {
            String recipientEmail = tenant.Email__c;
            String emailContent = '...'; // Reminder content
            Messaging.SingleEmailMessage email = new
Messaging.SingleEmailMessage();
            email.setToAddresses(new String[]{recipientEmail});
            email.setSubject('Reminder: Monthly Rent Payment Due');
            email.setPlainTextBody(emailContent);
            Messaging.sendEmail(new
Messaging.SingleEmailMessage[]{email});
        }
    }
}
```

Schedule:

- Apex Class: `MonthlyEmailScheduler`

- Frequency: Monthly
- Day: 1st of every month
- Start Time: 9:00 AM

Flow: Monthly Payment Acknowledgment

- Trigger: Record Updated (Payment for tenant)
- Condition: `check_for_payment__c = Paid`
- Action: Send Email
    - To: `{!$Record.Tenant__r.Email__c}`
    - Subject: "Confirmation of Successful Monthly Payment"
    - Body: From text template with tenant name

Testing

- Approval: Submit → Approve → Email + Notification sent
- Trigger: Try to assign same property to multiple tenants → Error shown
- Flow: Mark payment as Paid → Confirmation email sent
- Scheduler: Email sent on 1st of each month

Appendix

- **Objects Created**: Property, Tenant, Lease, Payment
- **Relationships**: Lookup and Master-Detail
- **Automation**: Triggers, Flows, Approval Process
- **Reports** (not covered in steps, but can be added):
    - Leases Expiring Soon
    - Overdue Payments
    - Tenants by Property

Conclusion

The Lease Management System is a crucial tool designed to simplify and automate the complex processes involved in managing lease agreements. By integrating features such as centralized record-keeping, automated notifications, compliance tracking, and streamlined communication, the system ensures greater efficiency, accuracy, and transparency. This project not only reduces administrative workload but also minimizes the risk of errors and legal issues. Overall, the successful implementation of this system supports better decision-making, improves operational efficiency, and adds significant value to the lease management process.