LEASE MANAGEMENT

NAME :Shaik Fayazuddin
EMAIL ID: shaikfayazuddin22@gmail.com
COLLEGE:CVR COLLEGE OF ENGINEERING

Project Overview

The **Lease Management System** is a robust Salesforce-based solution designed to automate and streamline all operations related to leasing real estate properties, equipment, or other assets. The goal is to enhance productivity by managing tenant details, tracking payments, monitoring lease life cycles, and sending automated alerts and approval requests.

Phase 1: Requirement Analysis & Planning

Objectives:

- Define and document end-to-end lease-related business processes.
- Identify key user roles and their access permissions.
- Determine core modules: **Property, Tenant, Lease, Payment**.
- Establish communication plans and define approval workflows.

Key Requirements:

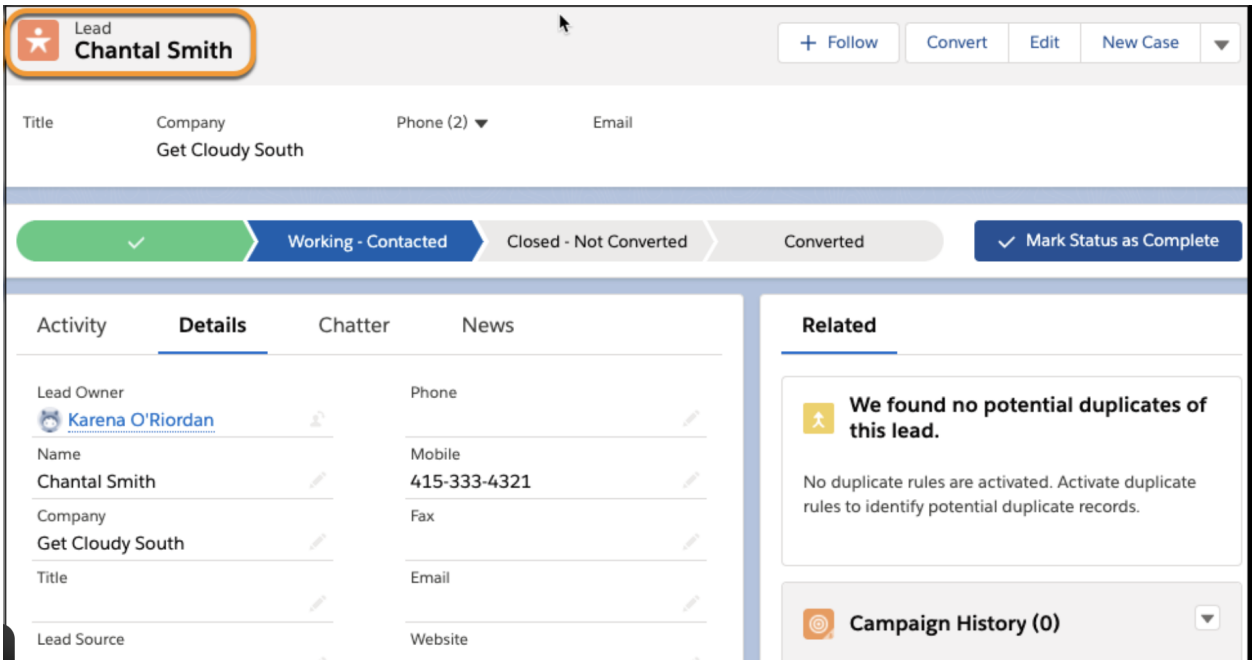
- Full **CRUD** operations for lease and related data.
- Establish relationships between **tenants** and **properties**.
- Track **monthly rent payments**.
- Send **email notifications**, approvals, and reminders.
- Apply **validation rules** for data consistency.
- Use **Apex triggers** for enforcing business logic.
- Schedule **monthly rent reminders** automatically.

Phase 2: Backend Setup in Salesforce

Custom Object Creation

1. **Property Object**
    - **Label:** Property
    - **Plural Label:** Properties
    - **Record Name:** Property Name (Text)
    - **Settings:** Enable Reports, Track Field History, Activities, Global 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:** Leases
    - **Record Name:** Lease Name (Text)



Phase 3: UI/UX Development & Customization

Tabs Creation

For each custom object:

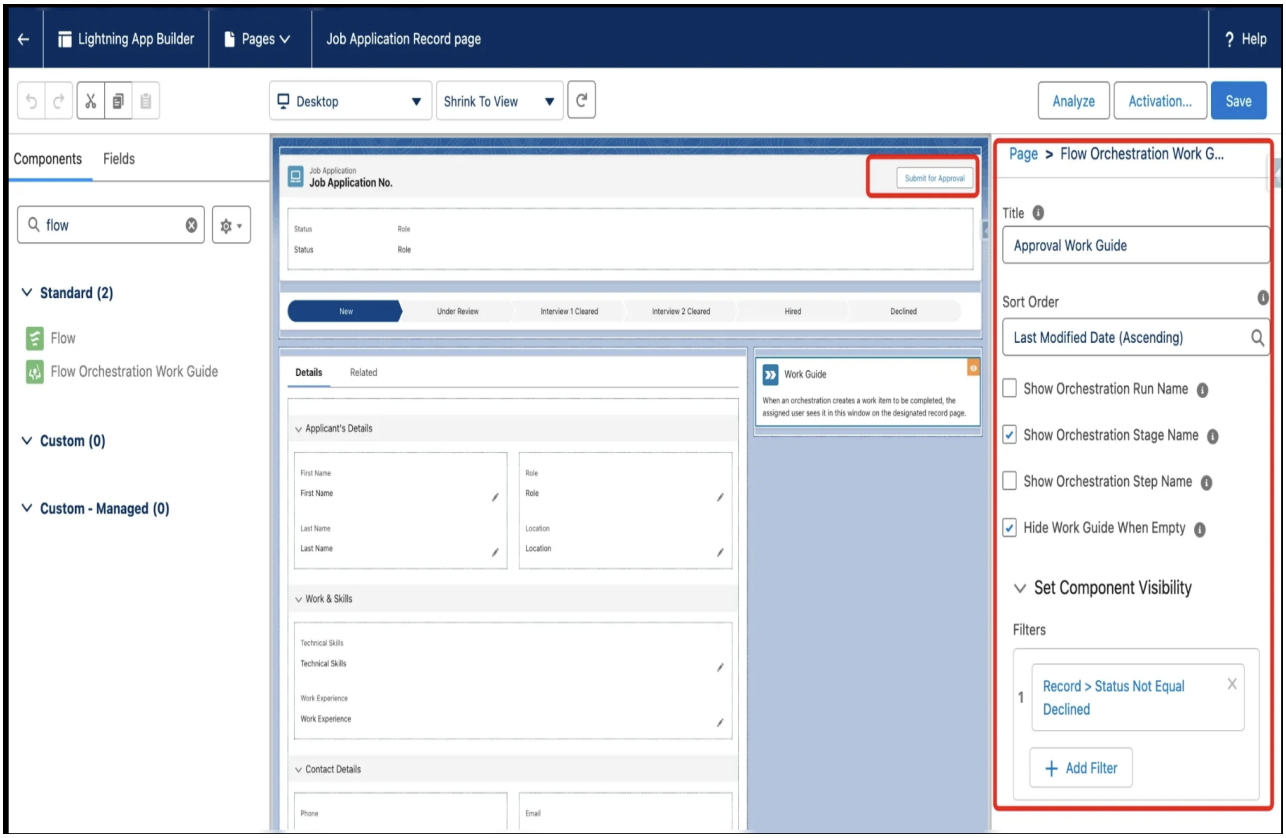
- Navigate to: **Setup → Tabs → New Custom Object Tab**
- Choose the object (e.g., Property)
- Select tab style
- Assign to user profiles (default)
- Check **"Append to users' personal customizations"**

Repeat this for:

- Property
- Tenant
- Lease
- Payment for Tenant

Lightning App Creation

1. Navigate 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**



Field Creation

Property Object

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

Tenant Object

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

Lease Object

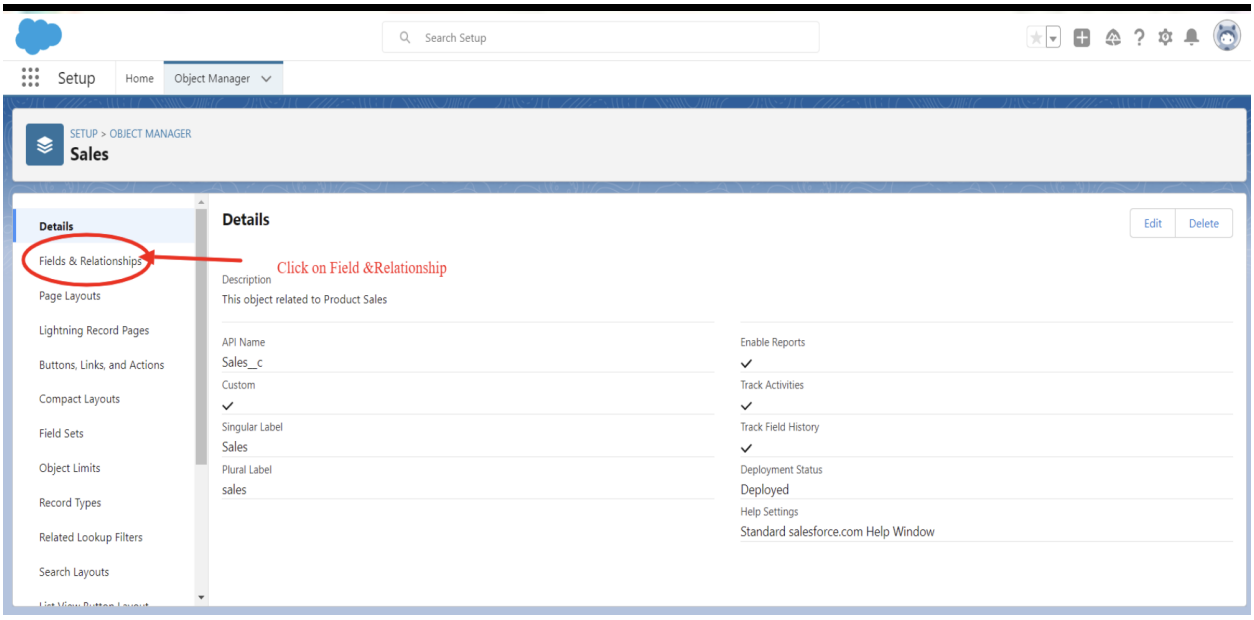
- **Start Date** (Date)
- **End Date** (Date)
- **Property** (Lookup to Property)

Payment for Tenant Object

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

Relationships

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



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:**

  `"End date must be greater than start date."`

Email Templates

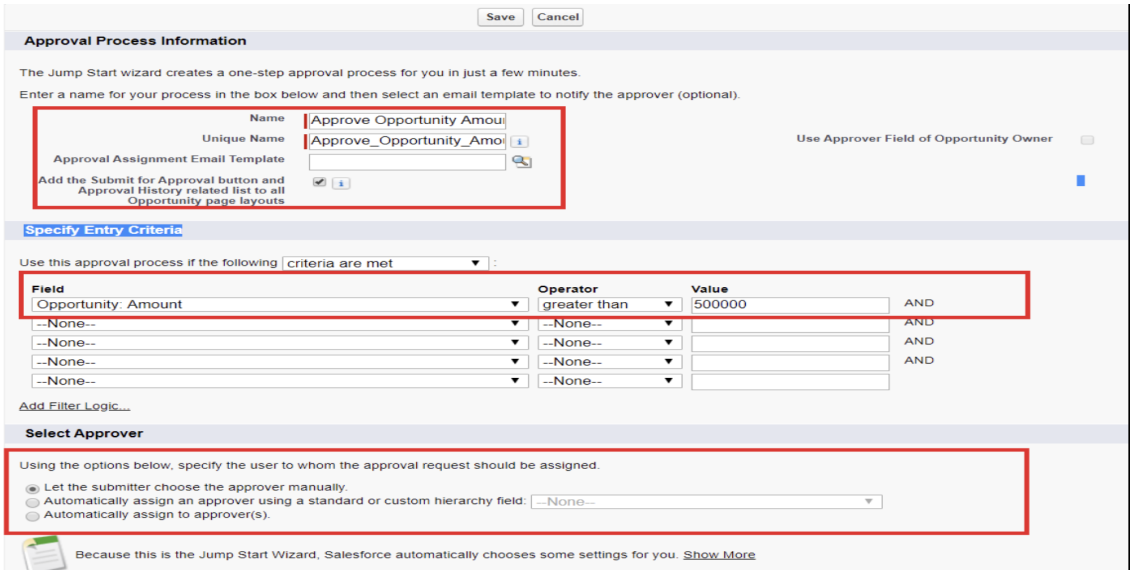
Template Name	Subject	Used In
Tenant Leaving	Request for approve the leave	Initial Approval
Leave Approved	Leave Approved	Final Approval
Leave Rejected	Leave Rejected	Final Rejection
Tenant Email	Monthly Rent Payment Reminder	Scheduler
Tenant Payment	Confirmation of Successful Payment	Flow Action

Approval Process – Check for Vacant

- **Object:** Tenant
- **Entry Criteria:** Status ≠ Leaving
- **Submitter:** Property Owner
- **Approver:** System Administrator

Approval Steps:

- **Initial Submission:** Send email → Tenant Leaving
- **If Approved:** Send email → Leave Approved
- **If Rejected:** Send email → Leave Rejected



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 emailContent = 'This is a reminder to pay your
monthly rent.';
            Messaging.SingleEmailMessage email = new
Messaging.SingleEmailMessage();
            email.setToAddresses(new String[]{tenant.Email__c});
            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
- **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:** Use a text template with dynamic tenant name

Testing

- **Approval Process:** Submit → Approve → Correct email sent
- **Trigger Validation:** Assign same property to multiple tenants → Error displayed
- **Flow:** Mark payment as Paid → Confirmation email sent
- **Scheduler:** Email is sent on the 1st of each month

Appendix

- **Objects Created:** Property, Tenant, Lease, Payment
- **Relationships:** Lookup and Master-Detail
- **Automation Used:**
    - Apex Triggers
    - Flow Builder
    - Approval Process
    - Scheduled Apex

Suggested Reports (Add-on Feature):

- Leases Expiring in Next 30 Days
- Overdue Rent Payments
- Tenant Distribution by Property Type

Conclusion

The **Lease Management System** offers an end-to-end solution for managing leasing operations with minimal manual intervention. Leveraging Salesforce's powerful automation tools, the system enhances accuracy, communication, and visibility into the leasing process. With automated rent reminders, validation checks, and approval flows, it significantly reduces administrative overhead and improves operational efficiency.