

# **LEASE MANAGEMENT**

**College Name: IDEAL INSTITUTE OF TECHNOLOGY**

**TEAM ID:** **LTVIP2025TMID30830**

**TEAM MEMBERS:**

**Team Leader: Harischandra vara prasad velaga**

**Email:** [Harivelaga8@gmail.com](mailto:Harivelaga8@gmail.com)

**Team Member: Areti chandrvyas**

**Email:** [Chandravyasareti14@gmail.com](mailto:Chandravyasareti14@gmail.com)

**Team Member: Kodali murali krishna**

**Email:** [Kodalimuralikrishna2@gmail.com](mailto:Kodalimuralikrishna2@gmail.com)

**Team Member: Pamulapati Ravindra**

**Email:** [228X1A4559@khitguntur.ac.in](mailto:228X1A4559@khitguntur.ac.in)

# 1. INTRODUCTION

## 1.1 Project Overview

The Lease Management System is a Salesforce-based application designed to streamline the processes associated with leasing real estate properties. It handles tenant management, lease contracts, payments, and communication with automation features such as flows, approval processes, and email alerts.



## 1.2 Purpose

The main objective of the project is to enable organizations to efficiently manage properties, tenants, and lease-related activities. It reduces manual intervention, improves accuracy, and ensures better compliance and communication.

## **2. IDEATION PHASE**

### **2.1 Problem Statement**

Managing lease information manually can lead to errors, missed payments, and communication delays. There is a need for a centralized, automated system to maintain lease records, track payments, and manage tenants efficiently.

### **2.2 Empathy Map Canvas**

- **Think & Feel:** Needs confidence in lease management.
- **Hear:** Concerns from tenants about payment reminders.
- **See:** Missed lease renewals and tracking issues.
- **Say & Do:** Wants a smooth system to manage leases and communications.
- **Pain:** Manual tracking, miscommunication.
- **Gain:** Streamlined processes, reduced errors.

### **2.3 Brainstorming**

Ideas included automated alerts, integrated payment tracking, custom objects for modular data, and use of approval processes to manage tenant transitions.

---

## **3. REQUIREMENT ANALYSIS**

### **3.1 Customer Journey Map**

Tenants interact with the system by applying for leases, receiving emails for payment and status updates, and submitting leave requests. Admins handle object creation, approval processes, and maintain oversight through dashboards and flows.

### **3.2 Solution Requirement**

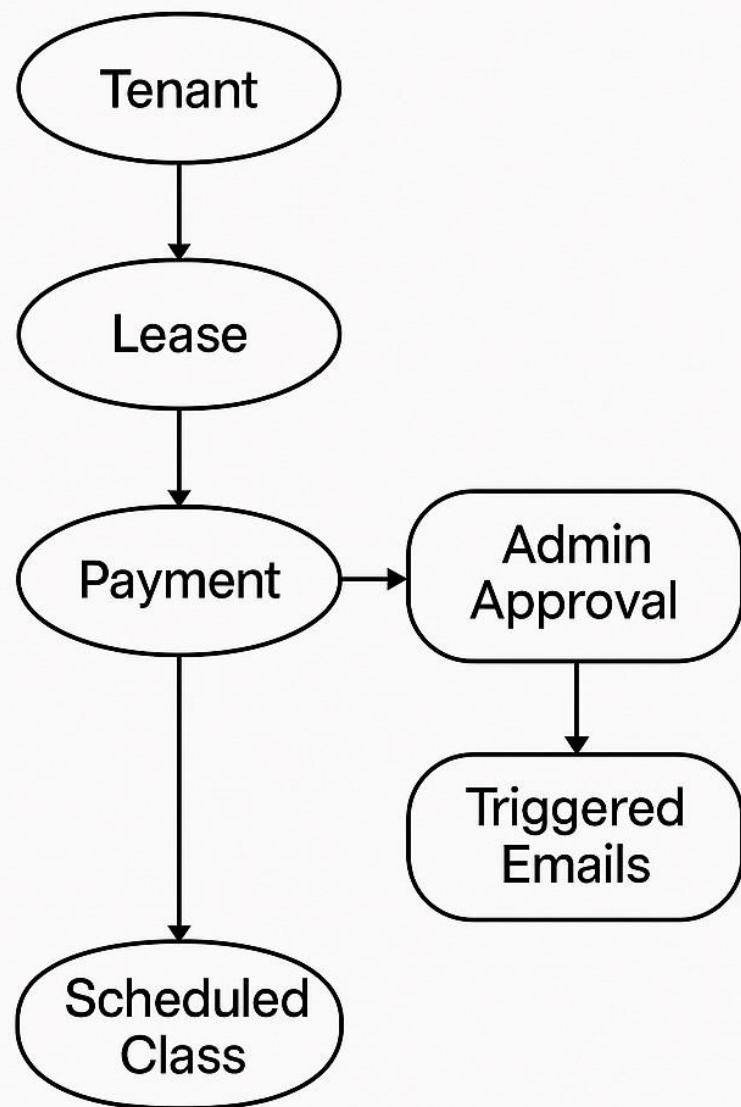
- Salesforce platform
- Custom Objects: Property, Tenant, Lease, Payment
- Email Templates
- Apex Triggers & Classes
- Flow automation
- Approval Processes

### 3.3 Data Flow Diagram

Tenant → Lease → Payment

Admin approval → Triggered Emails  
↳ Scheduled class → Monthly Reminders

### 3.3 Data Flow Diagram



## **3.4 Technology Stack**

- **Platform:** Salesforce Lightning
  - **Language:** Apex
  - **Automation:** Salesforce Flow
  - **Database:** Salesforce Custom Objects
  - **Communication:** Classic Email Templates
- 

# **4. PROJECT DESIGN**

## **4.1 Problem-Solution Fit**

The system solves lease tracking, tenant management, and payment notification problems using Salesforce's automation and customization features.

## **4.2 Proposed Solution**

Create custom Salesforce objects and implement workflows, email templates, and Apex triggers to automate all critical lease management functions.

## **4.3 Solution Architecture**

- **Frontend:** Salesforce UI via Lightning Apps
  - **Backend:** Custom Objects and Apex
  - **Automation:** Flows, Validation Rules, Scheduled Jobs
  - **Integration:** Email alerts via templates
- 

# **5. PROJECT PLANNING & SCHEDULING**

## **5.1 Project Planning**

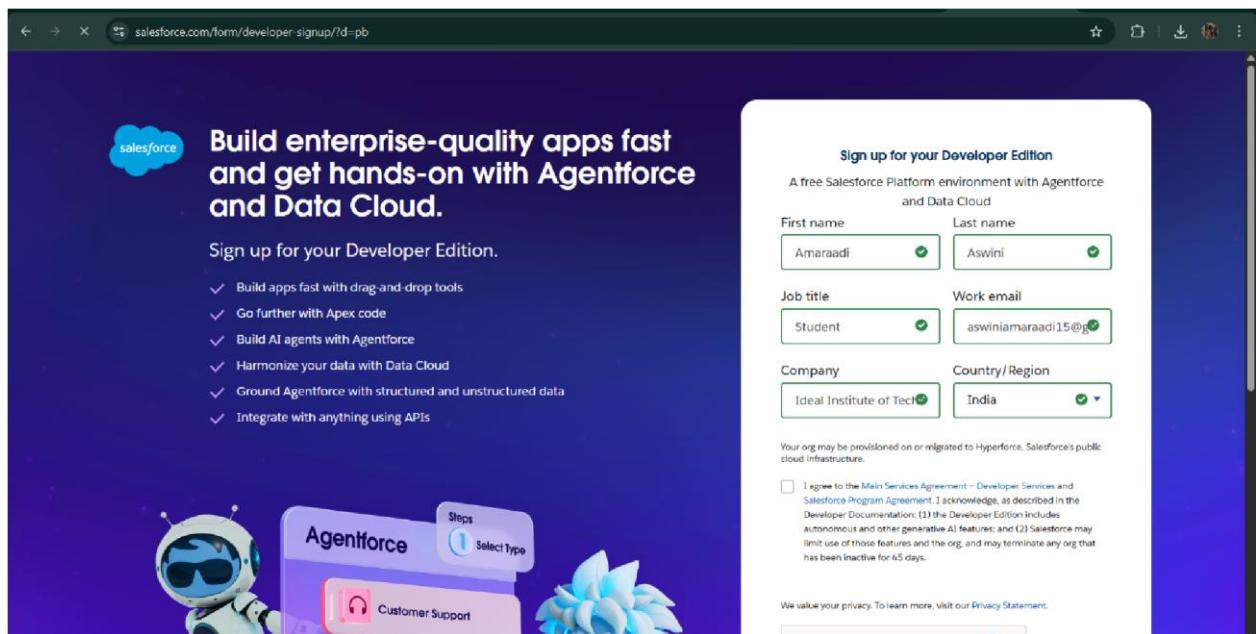
- Week 1: Object and Field Creation
- Week 2: Tab and App Setup
- Week 3: Flows and Email Templates

- Week 4: Apex Triggers & Approval Process
  - Week 5: Testing & Validation
- 

## 6. DEVELOPMENT PHASE

### Creating Developer Account:

By using this URL - <https://www.salesforce.com/form/developer-signup/?d=pb>



- Created objects: Property, Tenant, Lease, Payment

orgfarm-5df1e805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgk000000zTTI/Details/view

The screenshot shows the Salesforce Object Manager interface for the 'property' object. The left sidebar lists various configuration tabs: 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 'Details' tab is selected. The 'Fields & Relationships' section shows the API Name as 'property\_c', which is a Custom field. The Singular Label is 'property' and the Plural Label is also 'property'. The 'Enable Reports' section has checkboxes for Track Activities, Track Field History, and Deployment Status, all of which are checked. Help Settings and Standard salesforce.com Help Window are also listed.

orgfarm-5df1e805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgk000000zTbN/Details/view

The screenshot shows the Salesforce Object Manager interface for the 'Tenant' object. The left sidebar lists the same configuration tabs as the previous screenshot. The main 'Details' tab is selected. The 'Fields & Relationships' section shows the API Name as 'Tenant\_c', which is a Custom field. The Singular Label is 'Tenant' and the Plural Label is 'Tenants'. The 'Enable Reports' section has checkboxes for Track Activities, Track Field History, and Deployment Status, all of which are checked. Help Settings and Standard salesforce.com Help Window are also listed.

Setup > Object Manager

## lease

Details

Description

API Name: lease\_c  
Custom: ✓  
Singular Label: lease  
Plural Label: lease

Enable Reports: ✓  
Track Activities: ✓  
Track Field History: ✓  
Deployment Status: Deployed  
Help Settings: Standard salesforce.com Help Window

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

Edit Delete

Setup > Object Manager

## Payment for tenant

Details

Description

API Name: Payment\_for\_tenant\_c  
Custom: ✓  
Singular Label: Payment for tenant  
Plural Label: Payment

Enable Reports: ✓  
Track Activities: ✓  
Track Field History: ✓  
Deployment Status: Deployed  
Help Settings: Standard salesforce.com Help Window

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  
Scoping Rules

Edit Delete

- Configured fields and relationships

**Fields & Relationships**  
9 items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address__c	Long Text Area(32768)		
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Name	Name__c	Text(25)		
Owner	OwnerId	Lookup(User,Group)	✓	
property	property__c	Lookup(property)	✓	
property Name	Name	Text(30)	✓	
sfqt	sfqt__c	Text(18)		
Type	Type__c	Picklist		

**Fields & Relationships**  
7 items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Amount	Amount__c	Number(18, 0)		
check for payment	check_for_payment__c	Picklist		
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)	✓	
Payment date	Payment_date__c	Date		
Payment Name	Name	Text(80)	✓	

Setup > Object Manager

## lease

Fields & Relationships				
7 Items, Sorted by Field Label				
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedBy	Lookup(User)		
End date	End_date_c	Date		
Last Modified By	LastModifiedBy	Lookup(User)		
lease Name	Name	Text(80)		
Owner	OwnerId	Lookup(User,Group)		
property	property_c	Lookup(property)		
start date	start_date_c	Date		

Setup > Object Manager

## Tenant

Fields & Relationships				
7 Items, Sorted by Field Label				
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedBy	Lookup(User)		
Email	Email_c	Email		
Last Modified By	LastModifiedBy	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		
Phone	Phone_c	Phone		
status	status_c	Picklist		
Tenant Name	Name	Text(80)		

- Developed Lightning App with relevant tabs

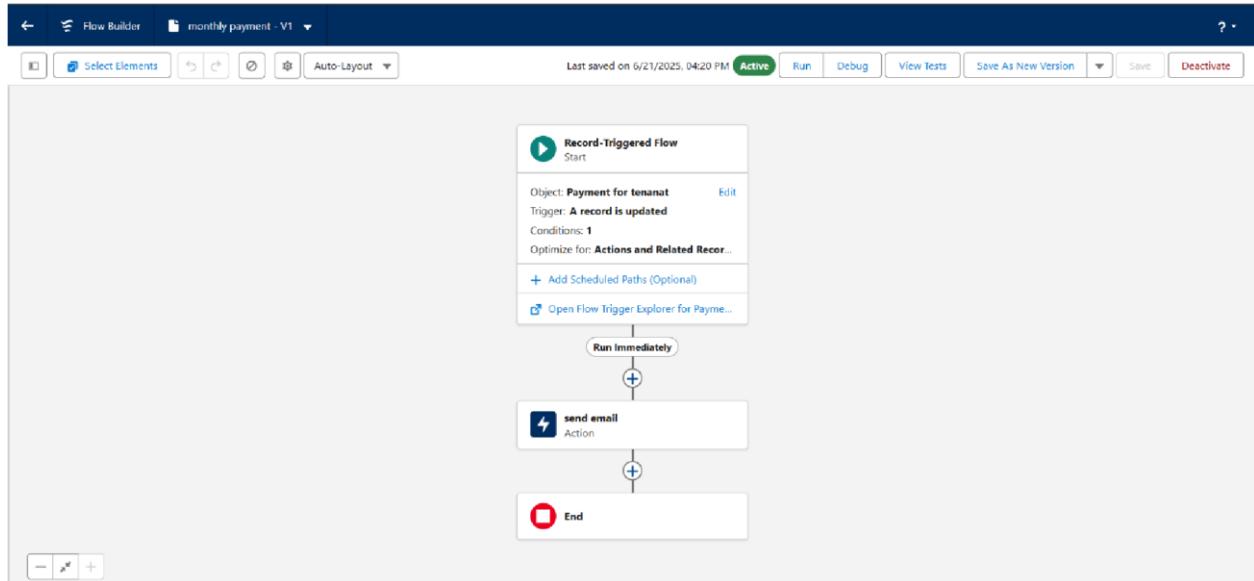
The screenshot shows the 'Lightning App Builder' interface with the 'App Details & Branding' tab selected. On the left, a sidebar lists 'App Options', 'Utility Items (Desktop Only)', 'Navigation Items', and 'User Profiles'. The main area contains fields for 'App Name' (Lease Management), 'Developer Name' (Lease\_Management), and a 'Description' box (Application to efficiently handle the processes related to lessing real estate properties). It also includes sections for 'App Branding' (with an image and primary color set to #0070D2) and 'Org Theme Options' (unchecked). A preview of the app launcher is shown at the bottom.

The screenshot shows the 'Lightning App Builder' interface with the 'Navigation Items' tab selected. The left sidebar includes 'App Details & Branding', 'App Options', 'Utility Items (Desktop Only)', and 'User Profiles'. The main area displays the 'Available Items' list (Accounts, Activation Targets, Activations, All Sites, Alternative Payment Methods, Analytics, App Launcher, Appointment Categories, Appointment Invitations, Approval Requests) and the 'Selected Items' list (Payment, Tenants, property, lease). Navigation arrows between the two lists allow items to be moved.

The screenshot shows the 'User Profiles' section of the Lightning App Builder. On the left, a sidebar lists 'App Settings' options: App Details & Branding, App Options, Utility Items (Desktop Only), Navigation Items, and User Profiles (which is currently selected). The main area is titled 'User Profiles' and contains a sub-instruction: 'Choose the user profiles that can access this app.' Below this is a 'Available Profiles' list containing various user profile names, each with a small preview icon. A search bar at the top of this list says 'Type to filter list...'. To the right, a 'Selected Profiles' panel shows a single profile named 'System Administrator' with a small preview icon. Navigation arrows (left and right) are positioned between the two panels.

The screenshot shows the 'Lease Management' interface, specifically the 'Payment' section. At the top, there's a navigation bar with tabs for 'Lease Management', 'Payment' (which is active), 'Tenants', 'property', and 'lease'. Below this is a search bar with placeholder text 'Search...' and a toolbar with icons for 'New', 'Import', 'Change Owner', and 'Assign Label'. The main content area is titled 'Recently Viewed' and displays a list of 5 items under 'Payment Name'. The list includes: 1. Rahul, 2. Jack, 3. Raj, 4. Sam, and 5. Lahari. Each item has a small checkbox icon to its left. To the right of the list is a column of small preview icons. At the bottom of the list, there are additional buttons for 'Search this list...', 'New', 'Import', 'Change Owner', 'Assign Label', and a 'More' button.

- Implemented Flows for monthly rent and payment success



- To create a validation rule to a Lease Object

The screenshot shows the Validation Rule Edit screen for the 'lease' object in the Setup interface:

- Validation Rule Edit**
- Rule Name:** lease\_end\_date
- Active:** checked
- Description:** (empty)
- Error Condition Formula:**

```
Example: Discount_Percent > 30 More Examples...
Display an error if Discount is more than 30%
If this formula expression is true, display the text defined in the Error Message area
```

`end_date__c < start_date__c`
- Functions:**
  - All Function Categories
  - ABS
  - ACOS
  - ADDMONTHS
  - AND
  - ASCII
  - ASIN

The screenshot shows the Salesforce Setup interface under the Object Manager. A validation rule for the 'lease' object is displayed. The validation rule is named 'lease\_end\_date' and has the formula 'End\_date\_\_c <= start\_date\_\_c'. The error message is 'Your End date must be greater than start date'. The rule is active and was created by the 'Sowmya Team' on June 19, 2025, at 5:37 AM, and last modified by the same team on June 26, 2025, at 7:47 AM.

- Added Apex trigger to restrict multiple tenants per property

The screenshot shows the 'Lease Management' application. A 'New Tenant' dialog is open, displaying fields for Tenant Name ('chinu'), Email ('chinu@gmail.com'), and Property ('Parkside'). An error message box is overlaid on the dialog, stating 'We hit a snag.' and 'Review the errors on this page.' with the note 'A tenant can have only one property.'

- Scheduled monthly reminder emails using Apex class

```

1 *global class MonthlyEmailScheduler implements Schedulable {
2
3     global void execute(SchedulableContext sc) {
4
5         Integer currentDay = Date.today().day();
6
7         if (currentDay == 1) {
8
9             sendMonthlyEmails();
10        }
11    }
12
13 }
14
15
16 *public static void sendMonthlyEmails() {
17
18     List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
19
20     for (Tenant__c tenant : tenants) {
21
22         String recipientEmail = tenant.Email__c;
23
24         String emailContent = 'I trust this email finds you well. I am writing to remind you that the monthly rent is due. Your timely payment ensures the smooth functioning of our rental arrangement and helps maintain';
25
26         String emailSubject = 'Reminder: Monthly Rent Payment Due';
27
28         Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
29
30         email.setToAddresses(new String[]{recipientEmail});
31
32         email.setSubject(emailSubject);
33
34         email.setPlainTextBody(emailContent);
35     }
36 }

```

- Built and tested email templates for leave request, approval, rejection, payment, and reminders

**Classic Email Templates**

**Email Template Detail**

Field	Value
Email Templates from Salesforce	Unified Public Classic Email Templates
Email Template Name	Leave approved
Template Unique Name	Leave_approved
Encoding	Unicode (UTF-8)
Author	Sowmya Team (Changed)
Created By	Sowmya Team 6/20/2025, 1:08 AM
Modified By	Sowmya Team 6/20/2025, 1:08 AM

**Email Template**

**Plain Text Preview**

```

dear({Tenant__c.Name}),

I hope this message finds you well. I am writing to inform you that I have received your email confirming the approval of my leave request. I would like to express my gratitude for considering and approving my time off.

Your leave is confirmed. You can leave now.

```

The screenshot shows the Salesforce Setup interface with the 'Email' tab selected. Under 'Email', 'Classic Email Templates' is highlighted. A search bar at the top left shows 'email template'. The main content area displays a 'Text Email Template' named 'tenant leaving'. The 'Email Template Detail' section includes fields like Email Template Name (tenant leaving), Template Unique Name (tenant\_leaving), Encoding (Unicode (UTF-8)), Author (Sowmya Team [Changed]), and Created By (Sowmya Team, 6/20/2025, 1:06 AM). The 'Available For Use' checkbox is checked. The 'Email Template' preview section shows a subject line 'request for approve the leave' and a plain text preview: 'Dear {Tenant\_\_c.CreatedBy}, Please approve my leave'.

This screenshot shows the same Salesforce Setup interface as the previous one, but the email template is now named 'Leave rejected'. The 'Email Template Detail' section shows the same details as the first template, including the same subject line and plain text preview. The preview text reads: 'Dear {Tenant\_\_c.Name}, I hope this email finds you well. Your contract has not ended. So we can't approve your leave. Your leave has rejected'.

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

- Page Header:** Search bar with "Search Setup", a gear icon, and other navigation icons.
- Left Sidebar:** "Setup" tab selected, "Email" category, "Classic Email Templates" selected under "Email".
- Search Bar:** "email template".
- Section:** "SETUP Classic Email Templates".
- Email Template Detail:**
  - Name:** Tenant Email
  - Template Unique Name:** Tenant\_Email
  - Encoding:** Unicode (UTF-8)
  - Author:** Scwmya\_Team [Change]
  - Created By:** Scwmya\_Team, 6/20/2025, 1:12 AM
  - Modified By:** Scwmya\_Team, 6/20/2025, 1:12 AM
- Email Template Preview:**
  - Subject:** Urgent: Monthly Rent Payment Reminder
  - Plain Text Preview:**

Dear {Tenant\_\_c\_Name}.

I trust this email finds you well. We appreciate your continued tenancy at our property and I hope you have been comfortable in your residence.

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

- Page Header:** Search bar with "Search Setup", a gear icon, and other navigation icons.
- Left Sidebar:** "Setup" tab selected, "Email" category, "Classic Email Templates" selected under "Email".
- Search Bar:** "email template".
- Section:** "SETUP Classic Email Templates".
- Email Template Detail:**
  - Name:** tenant payment
  - Template Unique Name:** tenant\_payment
  - Encoding:** Unicode (UTF-8)
  - Author:** Scwmya\_Team [Change]
  - Created By:** Scwmya\_Team, 6/20/2025, 1:13 AM
  - Modified By:** Scwmya\_Team, 6/20/2025, 1:13 AM
- Email Template Preview:**
  - Subject:** Confirmation of Successful Monthly Payment
  - Plain Text Preview:**

Dear {Tenant\_\_c\_Email\_\_c}.

We hope this email finds you well. We are writing to inform you that we have successfully received your monthly payment. Thank you for your prompt and diligent payment.

- Approval Process creation

For Tenant Leaving:

**Approval Processes**  
Tenant: TenantApproval  
Process Definition Detail

Process Name	TenantApproval	Active
Unique Name	TenantApproval	Next Automated Approver Determined By
Description		
Entry Criteria	Tenant: status EQUALS Stay	
Record Editability	Administrator ONLY	Allow Submitters to Recall Approval Requests
Approval Assignment Email Template		
Initial Submitters	Tenant Owner	Modified By
Created By	Sowmya_Team, 6/23/2025, 3:41 AM	Sowmya_Team, 6/26/2025, 11:57 PM

**Initial Submission Actions**

Action Type	Description
Record Lock	Lock the record from being edited

**Approval Steps**

Action	Step Number	Name	Description	Criteria	Assigned Approver	Reject Behavior
Show Actions   Edit	1	Step 1			User:Sowmya_Team	Final Rejection

For Check for Vacant:

**Approval Processes**  
Tenant: check for vacant  
Process Definition Detail

Process Name	check for vacant	Active
Unique Name	check_for_vacant	Next Automated Approver Determined By
Description		
Entry Criteria	Tenant: status EQUALS Leaving	
Record Editability	Administrator ONLY	Allow Submitters to Recall Approval Requests
Approval Assignment Email Template	Leave approved	
Initial Submitters	Tenant Owner	Modified By
Created By	Sowmya_Team, 6/20/2025, 3:18 AM	Sowmya_Team, 6/26/2025, 11:02 PM

**Initial Submission Actions**

Action	Type	Description
Record Lock		Lock the record from being edited
Edit   Remove	Email Alert	please approve.my.leave

**Approval Steps**

Action	Step Number	Name	Description	Criteria	Assigned Approver	Reject Behavior
Show Actions   Edit	1	step1			User:Sowmya_Team	Final Rejection

- Apex Trigger

Create an Apex Trigger

Screenshot of the Salesforce IDE showing the Apex code for a trigger and a modal dialog.

```

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

```

The modal dialog "Open" shows the selected entity type is "Triggers".

Entity Type	Entities	Related
Classes	test	
Triggers		
Pages		
Page Components		
Objects		
Static Resources		
Packages		

Logs Tests Checkpoints Query Editor View State Progress Problems

Developer Console - Google Chrome

File Edit Debug Test Workspace Help < >

test.apex testHandler.apfc MonthlyEmailScheduler.apfc

Code Coverage: None API Version: 64 Go To

```

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

```

Logs Tests Checkpoints Query Editor View State Progress Problems

Create an Apex Handler class

Developer Console - Google Chrome

File Edit Debug Test Workspace Help < >

testHandler.apc MonthlyEmailScheduler.apc

Code Coverage: None API Version: 64 Go To

```

1 * public class testHandler {
2
3     public static void preventInsert(List<Tenant__c> newList) {
4
5         Set<Id> existingPropertyIds = new Set<Id>();
6
7         for (Tenant__c existingTenant : [SELECT Id, Property__c FROM Tenant__c WHERE Property__c != null]) {
8
9             existingPropertyIds.add(existingTenant.Property__c);
10
11         }
12
13         for (Tenant__c newTenant : newList) {
14
15             if (newTenant.Property__c != null) {
16
17                 newTenantaddError('A tenant can have only one property');
18             }
19
20         }
21
22     }
23

```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

Entity Type Entity Name Namespace Related

Entity Type	Entity Name	Namespace	Related
Entitiy type	testHandler	MonthlyEmailScheduler	↪ test ApexTrigger References ↪ property CustomField References ↪ Tenant__c SObject References ↪ Tenant__c SObject References
Classes			
Triggers			
Pages			
Page Components			
Objects			
Static Resources			
Packages			

Open Filter: Filter the repository (\* = any string) Hide Managed Packages Refresh

Developer Console - Google Chrome

File Edit Debug Test Workspace Help < >

testHandler.apc MonthlyEmailScheduler.apc

Code Coverage: None API Version: 64 Go To

```

1 * public class testHandler {
2
3     public static void preventInsert(List<Tenant__c> newList) {
4
5         Set<Id> existingPropertyIds = new Set<Id>();
6
7         for (Tenant__c existingTenant : [SELECT Id, Property__c FROM Tenant__c WHERE Property__c != null]) {
8
9             existingPropertyIds.add(existingTenant.Property__c);
10
11         }
12
13         for (Tenant__c newTenant : newList) {
14
15             if (newTenant.Property__c != null && existingPropertyIds.contains(newTenant.Property__c)) {
16
17                 newTenantaddError('A tenant can have only one property');
18             }
19
20         }
21
22     }
23

```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

## ● FLOWS

**Flow Builder** monthly payment - V1

Last saved on 6/21/2025, 04:20 PM Active Run Debug View Tests Save As New Version Save Deactivate

The screenshot shows two separate configurations for a "Record-Triggered Flow" named "monthly payment - V1".

**Configuration 1 (Top):**

- Start:** Triggered by "A record is updated" on the "Payment for tenantat" object.
- Condition:** Field "check for payment" Equals "Paid".
- When to Run the Flow for Updated Records:** Every time a record is updated and meets the condition requirements.
- Optimize Flow:** Set to "Actions and Related Records".
- Flow Steps:**
  - Run Immediately
  - Action: send email
  - End

**Configuration 2 (Bottom):**

- Start:** Triggered by "A record is updated" on the "Payment for tenantat" object.
- Configure Start:**
  - Select Object:** Payment for tenantat
  - Configure Trigger:** Trigger the Flow When: A record is updated
  - Set Entry Conditions:** All Conditions Are Met (AND)
- Flow Steps:**
  - Run Immediately
  - Action: send email
  - End

- Schedule class:  
Create an Apex Class

Developer Console - Google Chrome

orgfarm-5dfe805f2-dev-ed-develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File Edit Debug Test Workspace Help testHandler.apc MonthlyEmailScheduler.apc

Code Coverage Name API Version 64 Go To

```

1 * global class MonthlyEmailScheduler implements Schedulable {
2
3     global void execute(SchedulableContext sc) {
4
5         Integer currentDay = Date.today().day();
6
7         if (currentDay == 1) {
8
9             sendMonthlyEmails();
10
11        }
12
13    }
14
15
16    public static void sendMonthlyEmail()
17    {
18        List<Tenant__c> tenants = [SELECT
19
20            for (Tenant__c tenant : tenants
21
22                String recipientEmail = tenant.Email__c;

```

Open Entity Type Entities Related

Entity Type	Name	Namespace	Name	Extent	Direction
Classes	testHandler		MonthlyEmailScheduler		CustomField... References
Triggers			Email	CustomField... References	
Pages			<- Tenant__c	Subject	References
Page Components			<- Tenant__c	Subject	References
Objects					
Static Resources					
Packages					

Logs Tests Checkpoints Query Editor View Status Progress Problems

50% Reset

```

1 * global class MonthlyEmailScheduler implements Schedulable {
2
3     global void execute(SchedulableContext sc) {
4
5         Integer currentDay = Date.today().day();
6
7         if (currentDay == 1) {
8
9             sendMonthlyEmails();
10
11        }
12
13    }
14
15
16    public static void sendMonthlyEmail() {
17
18        List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
19
20        for (Tenant__c tenant : tenants) {
21
22            String recipientEmail = tenant.Email__c;
23
24            String emailContent = 'I trust this email finds you well. I am writing to remind you that the monthly rent is due your timely payment ensures the smooth functioning of our rental arrangement and helps maintain a positive living environment for all.';
25
26            String emailSubject = 'Reminder: monthly rent payment due';
27
28            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
29
30            email.setToAddresses(new String[]{recipientEmail});
31
32            email.setSubject(emailSubject);
33
34            email.setPlainTextBody(emailContent);
35
36            Messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
37
38        }
39
40    }
41
42 }

```

Logs Tests Checkpoints Query Editor View Status Progress Problems

## Schedule Apex class

SFDC Setup - Apex Classes

Search Setup

Apex Classes

Apex Class Detail

MonthlyEmailScheduler

Apex Class Detail

Name: MonthlyEmailScheduler

Namespace Prefix:

Created By: Sowmya Team, 6/23/2025, 2:46 AM

Status: Active

Code Coverage: 0% (0/15)

Last Modified By: Sowmya Team, 6/23/2025, 2:47 AM

Class Body

```
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) {  
    }  
}
```

Lease Management

Tenants

property

lease

Tenant: Aswini

Related Details

\* Tenant Name: Aswini

Owner: Sowmya Team

Email: aswiniamaraadi15@gmail.com

Phone: (905) 223-5567

status: Leaving

property: Imran

Created By: Sowmya Team, 6/26/2025, 6:05 AM

Last Modified By: Sowmya Team, 6/26/2025, 11:06 PM

Activity

New Case

New Lead

Delete

Clone

Change Owner

Printable View

Submit for Approval

Edit Labels

No activities to show.

No past activity. Past meetings and tasks marked as done show up here.

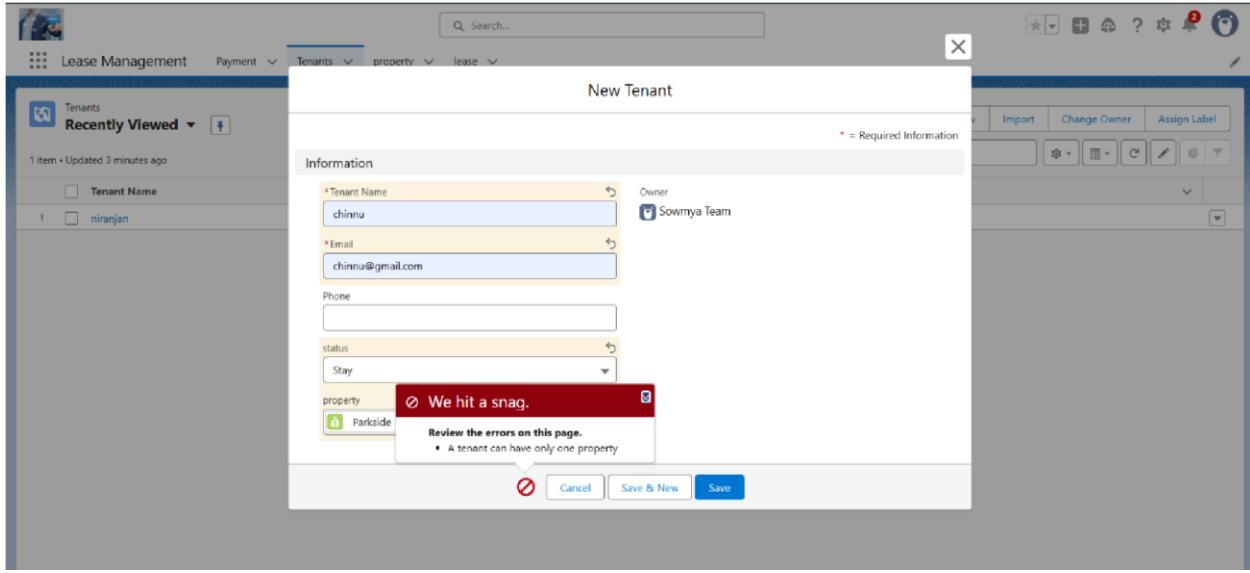
The screenshot shows a Salesforce Lightning interface for 'Lease Management'. A green banner at the top right indicates 'Tenant was submitted for approval.' The main area displays a 'Details' tab for a tenant named 'Aswini'. The form includes fields for Tenant Name (Aswini), Email (aswiniamaraadi15@gmail.com), Phone ((905) 223-5567), Status (Leaving), and Property (Imran). The owner is listed as 'Sowmya Team'. The bottom right of the form shows 'Last Modified By' and 'Last Modified Date'. Below the form are 'Cancel' and 'Save' buttons. To the right is an 'Activity' sidebar with sections for 'Upcoming & Overdue' and 'No past activity. Past meetings and tasks marked as done show up here.' The system status bar at the bottom shows weather (30°C Cloudy), a search bar, and various application icons.

The screenshot shows a Salesforce Lightning interface for a 'Process Instance Step' titled 'Tenant Approval'. The status is 'Approved'. The details section shows the submitter as 'Sowmya Team', date submitted as 'Jun 27, 2025', actual approver as 'Sowmya Team', and assigned to 'Sowmya Team'. Below this is an 'Approval Details' section with fields for Tenant Name (Aswini), Email (aswiniamaraadi15@gmail.com), and Property (Imran). To the right, a 'Notifications' sidebar lists five notifications all from 'Aswini' about approval requests, with timestamps ranging from 'a few seconds ago' to '19 hours ago'. The system status bar at the bottom shows weather (30°C Cloudy), a search bar, and various application icons.

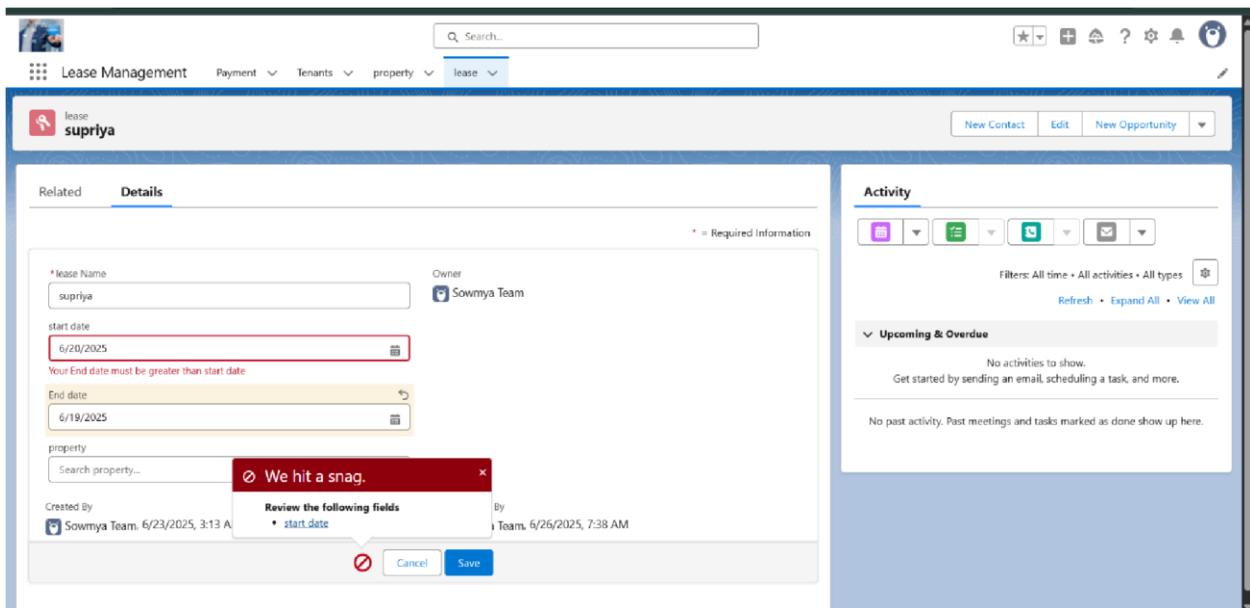
# 7. FUNCTIONAL AND PERFORMANCE TESTING

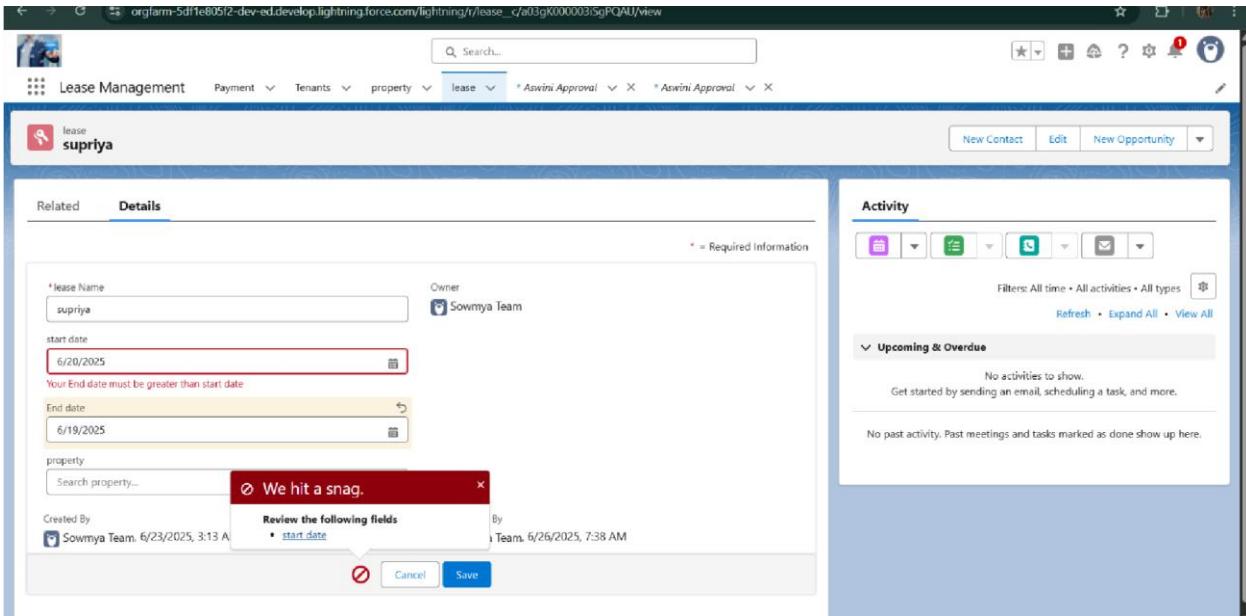
## 7.1 Performance Testing

- Trigger validation by entering duplicate tenant-property records

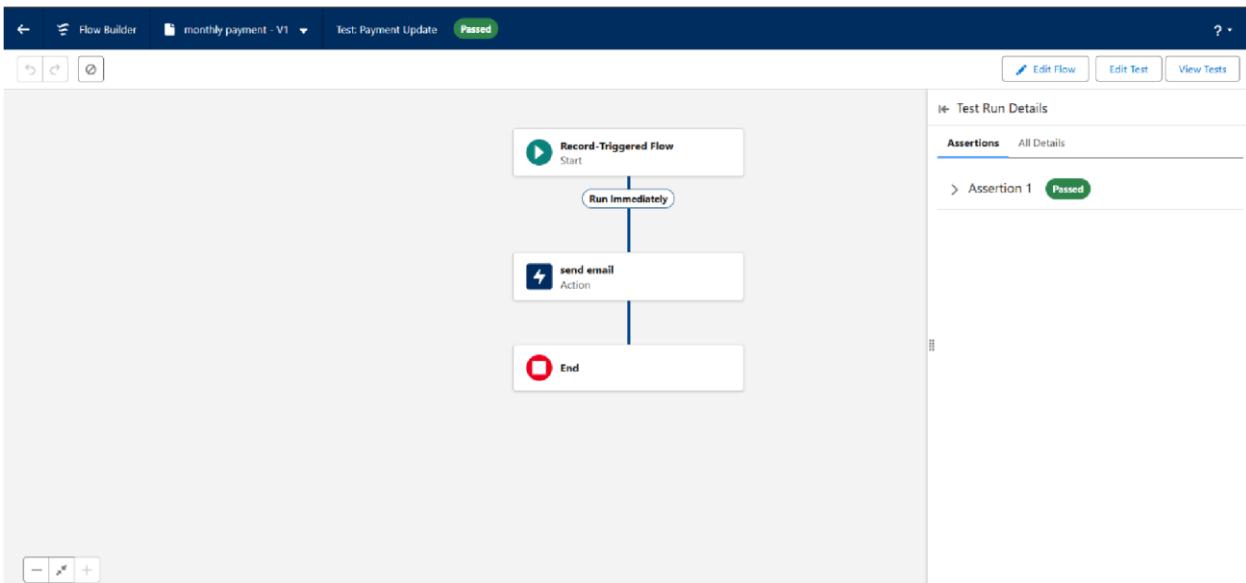


- Validation Rule checking





- Test flows on payment update



- Approval process validated through email alerts and status updates

The screenshot shows the Microsoft Dynamics 365 Lease Management interface. On the left, a 'Tenant' record for 'niranjan' is displayed in the 'Lease Management' module. The 'Details' tab is selected, showing fields for Tenant Name (niranjan), Email (niranjan1506@gmail.com), Phone, status (Stay), and property (Parkside Lofts). The record was created by 'Sowmya Team' on 6/23/2025 at 2:33 AM and last modified by 'Sowmya Team' on 6/23/2025 at 3:58 AM. On the right, a 'Notifications' sidebar lists several items:

- Approval request for the tenant is approved** (niranjan) - a few seconds ago
- Approval request for the tenant is rejected** (niranjan) - Jun 23, 2025, 4:29 PM
- Approval request for the tenant is approved** (niranjan) - Jun 23, 2025, 4:25 PM
- Approval request for the tenant is approved** (niranjan) - Jun 23, 2025, 4:14 PM
- New Guidance Center learning resource available** - Define Your Sales Process. Learn how to guide reps through the sales process. Jun 20, 2025, 1:28 PM

The screenshot shows the Microsoft Dynamics 365 Lease Management interface. On the left, the 'Approval History' section displays a table of steps:

Step Name	Date	Status	Assigned To
Step 1	6/25/2025, 5:39 AM	Approved	Sowmya Team
Approval Request Submitted	6/25/2025, 5:39 AM	Submitted	Sowmya Team
Step 1	6/23/2025, 3:59 AM	Rejected	Sowmya Team
Approval Request Submitted	6/23/2025, 3:58 AM	Submitted	Sowmya Team
Step 1	6/23/2025, 3:55 AM	Approved	Sowmya Team
Approval Request Submitted	6/23/2025, 3:55 AM	Submitted	Sowmya Team

Below the approval history is a 'Payment' section showing 2 entries: 'Jack' and 'Rahul'. A sidebar on the right provides a summary of recent activity.

# 8. RESULTS

## 8.1 Output Screenshots

- Tabs for Property, Tenant, Lease, Payment

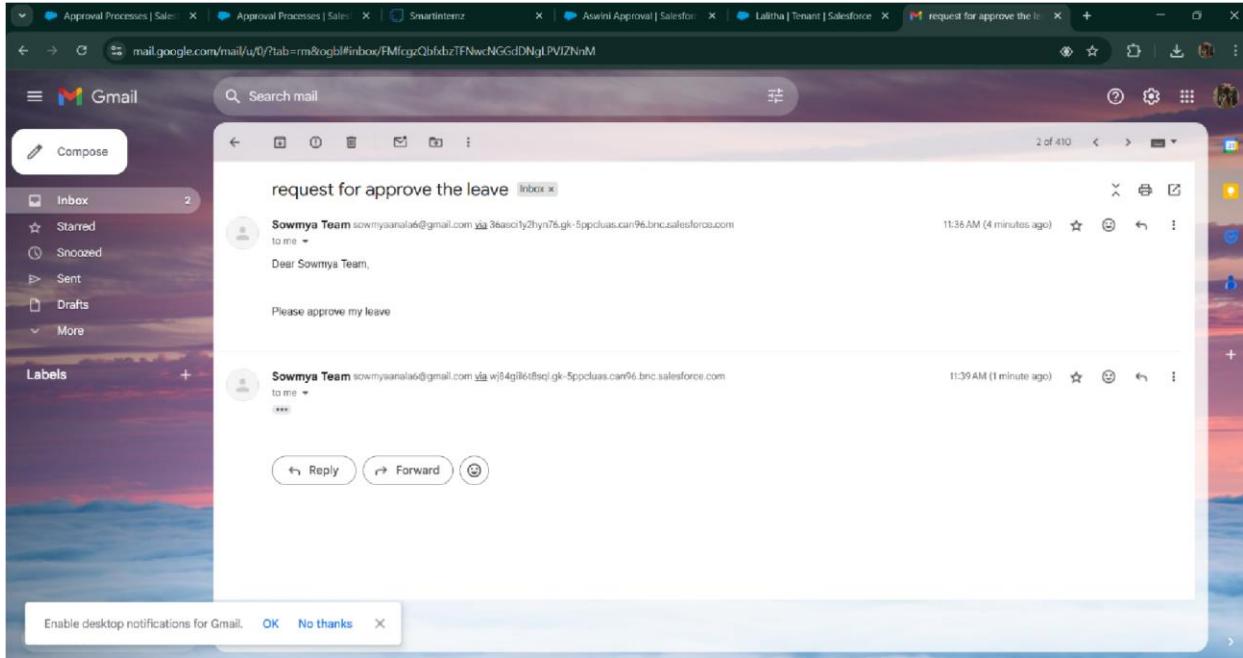
The screenshot shows the Salesforce Setup interface with the URL <https://orgfarm-5df1e805f2-dev-ed.lightning.force.com/lightning/setup/CustomTabs/page?address=%2Fsetup%2Fu%2Fcustomtabs.jsp%3FsetupId%3DCustomTabs%26retURL%3D%252Fsetu...>. The page title is "Custom Tabs". It displays three sections: "Custom Object Tabs", "Web Tabs", and "Visualforce Tabs". The "Custom Object Tabs" section lists four tabs: "Lease" (Tab Style: Keys), "Payment" (Tab Style: Credit Card), "Property" (Tab Style: Sack), and "Tenants" (Tab Style: Map). The "Web Tabs" and "Visualforce Tabs" sections both indicate "No [Type] Tabs have been defined".

- Email alerts

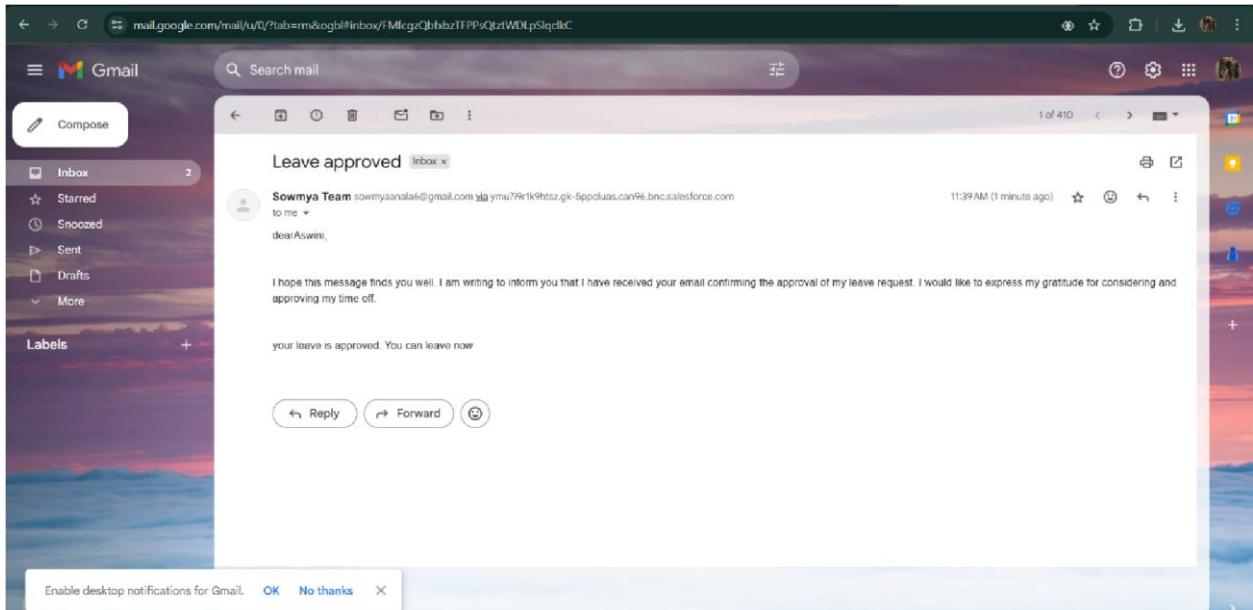
The screenshot shows the "Lease Management" application interface. The top navigation bar includes "Lease Management", "Payment", "Tenants", "property", and "lease". The main content area is titled "Approval History" and shows a table of approval steps for tenant "niranjan". The table has columns: Step Name, Date, Status, Assigned To, Actual Approver, and Comments. The steps listed are: Step 1 (Approved, Sowmya Team, Sowmya Team, approved), Approval Request Submitted (Submitted, Sowmya Team, Sowmya Team, leaving), Step 1 (Rejected, Sowmya Team, Sowmya Team, Rejected), Approval Request Submitted (Submitted, Sowmya Team, Sowmya Team, Leaving), Step 1 (Approved, Sowmya Team, Sowmya Team, Approved), Approval Request Submitted (Submitted, Sowmya Team, Sowmya Team, leaving), Step 1 (Approved, Sowmya Team, Sowmya Team, Approval Approved), and Approval Request Submitted (Submitted, Sowmya Team, Sowmya Team, Leaving).

Step Name	Date	Status	Assigned To	Actual Approver	Comments
1 Step 1	6/25/2025, 5:39 AM	Approved	Sowmya Team	Sowmya Team	approved
2 Approval Request Submitted	6/25/2025, 5:39 AM	Submitted	Sowmya Team	Sowmya Team	leaving
3 Step 1	6/23/2025, 3:59 AM	Rejected	Sowmya Team	Sowmya Team	Rejected
4 Approval Request Submitted	6/23/2025, 3:58 AM	Submitted	Sowmya Team	Sowmya Team	Leaving
5 Step 1	6/23/2025, 3:55 AM	Approved	Sowmya Team	Sowmya Team	Approved
6 Approval Request Submitted	6/23/2025, 3:55 AM	Submitted	Sowmya Team	Sowmya Team	leaving
7 Step 1	6/23/2025, 3:44 AM	Approved	Sowmya Team	Sowmya Team	Approval Approved
8 Approval Request Submitted	6/23/2025, 3:42 AM	Submitted	Sowmya Team	Sowmya Team	Leaving

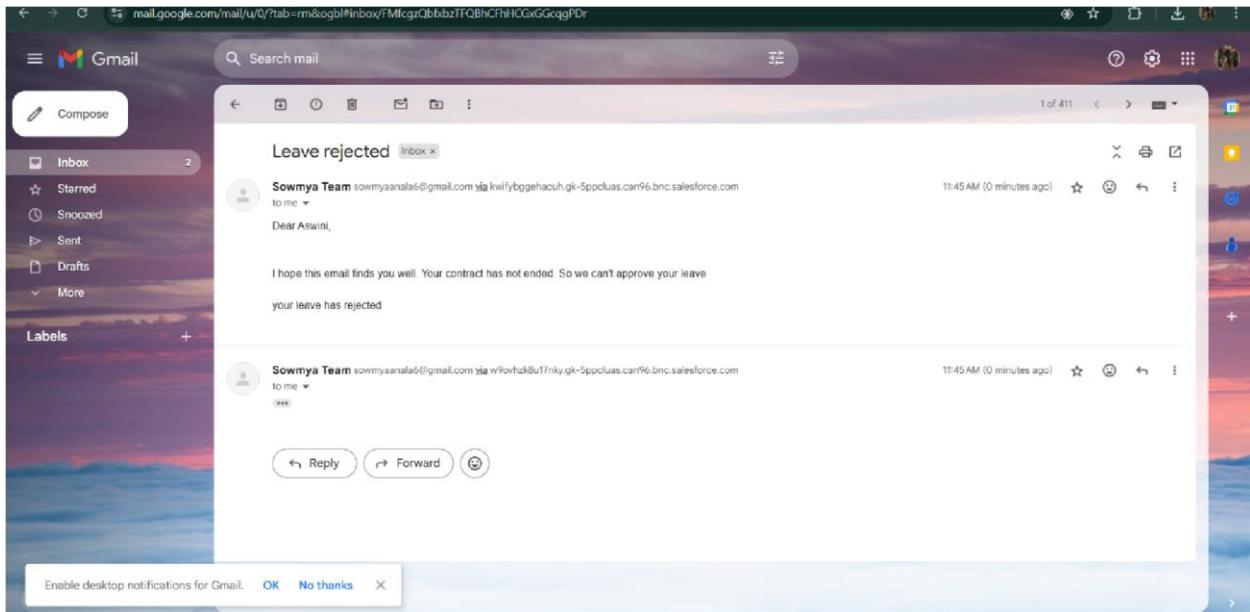
- Request for approve the leave



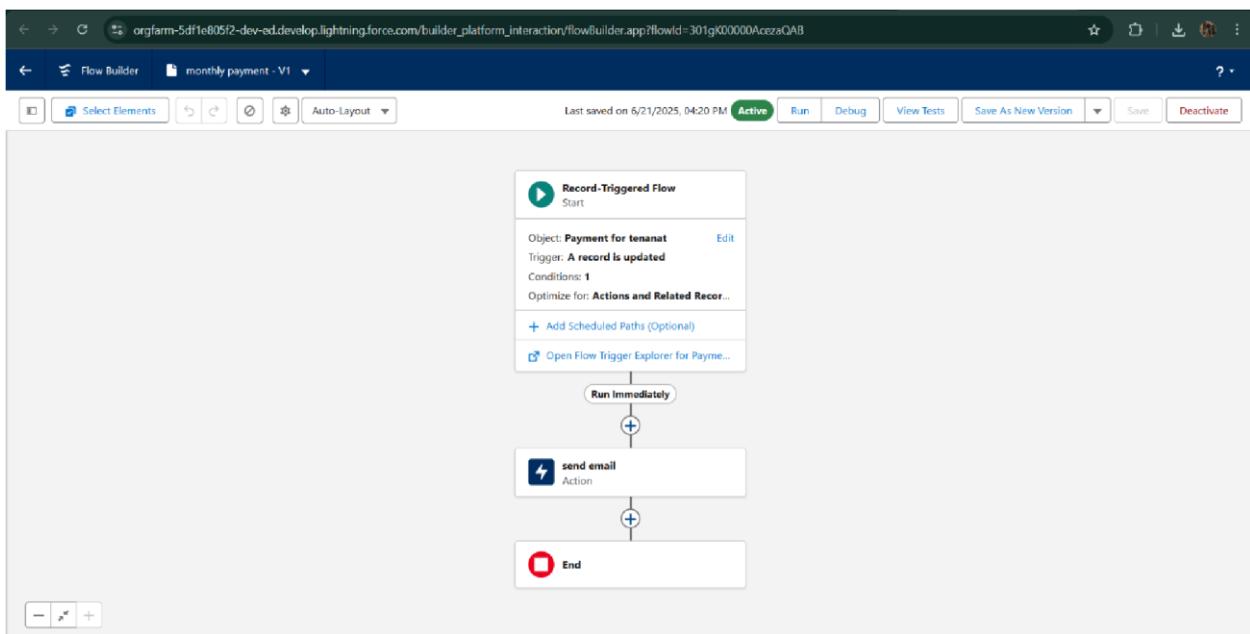
- Leave approved



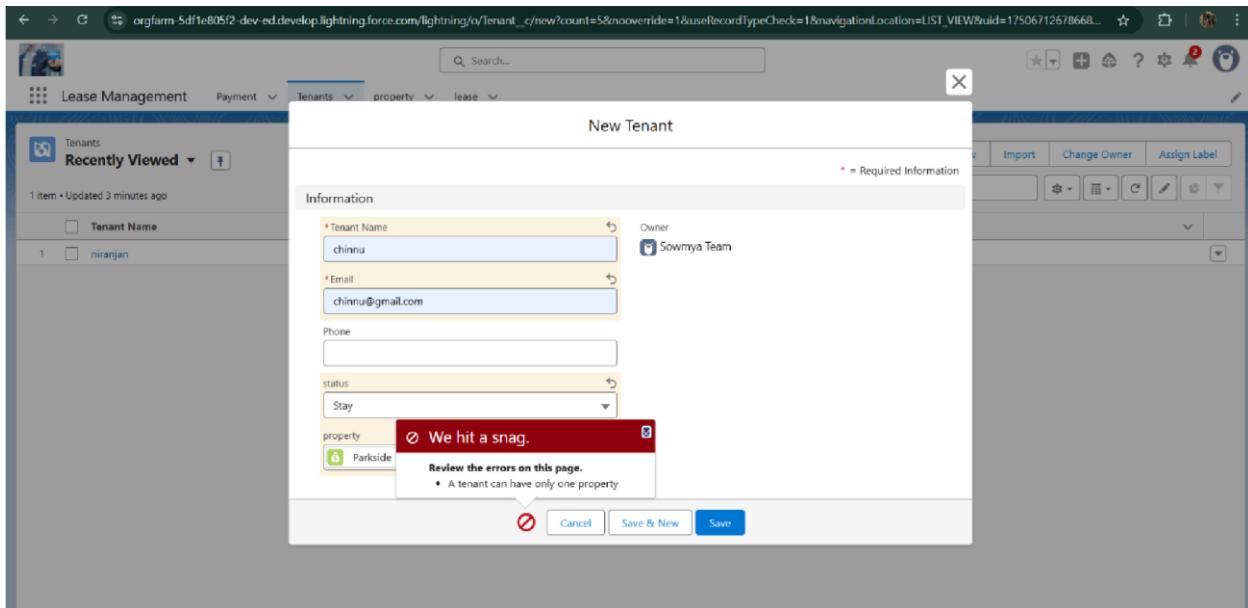
- Leave rejected



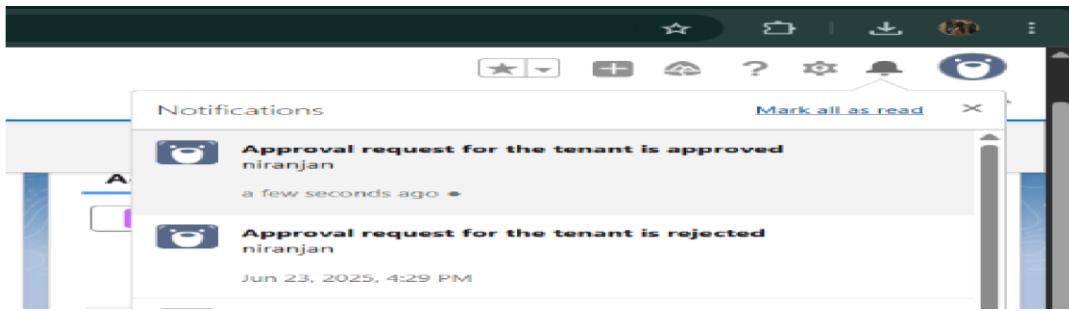
- Flow runs



- Trigger error messages



- Approval process notifications



## 9. ADVANTAGES & DISADVANTAGES

### Advantages:

#### 1. Centralized lease data

All lease agreements, documents, and tenant information are stored in one system, improving organization and accessibility.

#### 2. Automated lease tracking

System-generated reminders for lease renewals, rent due dates, and compliance deadlines reduce missed tasks and penalties.

### **3. Improved reporting and analytics**

Real-time dashboards and reports provide insights into occupancy, lease expirations, and revenue, supporting better decision-making.

### **4. Enhanced tenant communication**

Automated emails and notifications streamline communication with tenants about payments, renewals, or maintenance updates.

### **5. Audit readiness and compliance**

Accurate records and change tracking help ensure compliance with lease accounting standards and audit requirements.

---

### **Disadvantages:**

#### **1. Initial setup complexity**

Implementing a lease management system can be time-consuming, requiring detailed data migration and configuration.

#### **2. User training required**

Property managers and staff may need training to fully understand and use the system effectively.

#### **3. High implementation costs**

Custom development, integration, and licensing fees can be expensive, especially for smaller firms.

#### **4. Dependence on digital access**

Lease data is typically cloud-based, so system access requires internet connectivity and appropriate user permissions.

#### **5. Data security concerns**

Storing sensitive lease and financial data digitally introduces the risk of unauthorized access or data breaches if not properly secured.

---

## **10. CONCLUSION**

The Lease Management System successfully streamlines the operations of leasing through a structured, automated Salesforce application. It improves efficiency, communication, and data accuracy for both admins and tenants.

---

# 11. FUTURE SCOPE

## Future Scope of Lease Management System

### 1. Integration with Online Payment Gateways

Integrating secure online payment gateways (like Stripe, Razorpay, or PayPal) will allow tenants to pay rent, deposits, and other charges directly through the system. This ensures faster transactions, automated receipts, and real-time payment tracking—reducing manual reconciliation for finance teams.

### 2. Visual Dashboards for Better Insights

Advanced dashboards using tools like Salesforce Reports or third-party analytics platforms can provide real-time insights into lease status, occupancy rates, rent collection, pending renewals, and more. These visualizations help property managers make data-driven decisions and forecast trends effectively.

### 3. SMS Notification Features

Alongside emails, SMS alerts can notify tenants and staff about payment due dates, lease renewals, maintenance schedules, or compliance reminders. SMS communication ensures broader reach and immediate attention, especially when tenants are offline or prefer mobile updates.

### 4. Role-Based Access and Analytics

Implementing detailed role-based permissions allows specific users (e.g., admin, property manager, tenant) to access only relevant features and data. Coupling this with role-specific analytics helps each user view KPIs tailored to their responsibilities, improving operational efficiency and security.

### 5. Mobile App Support

A dedicated mobile app or mobile-responsive interface can empower tenants and managers to manage leases, payments, documents, and communications on the go. This improves accessibility and user engagement.

### 6. AI-Powered Predictive Maintenance

Future versions could use AI to analyse maintenance patterns and predict when certain property assets (like HVAC or plumbing systems) may require servicing—helping reduce downtime and costs.

### 7. E-signature and Document Automation

Integration with e-signature platforms (e.g., DocuSign, Adobe Sign) will allow digital signing of lease agreements and auto-generation of documents, streamlining onboarding and renewals.

## **8. Compliance and Legal Tracking**

Future features could include automated compliance tracking based on local real estate laws, lease accounting standards (like IFRS 16), and tax implications—minimizing legal risks.

## **9. IoT Integration for Smart Leasing**

Long-term, the system can integrate with IoT devices (e.g., smart locks, energy meters) for automated check-ins/out, usage-based billing, and enhanced security or environmental monitoring.

## **10. Multi-property and Franchise Support**

As property portfolios grow, the system can evolve to support multiple properties across regions, including different currencies, tax rules, and regulatory compliance—making it suitable for franchises or large real estate chains.

---

# **12. APPENDIX**

- **Source Code:** Provided in Apex Classes and Triggers

### **Test.apxt:**

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

### **testHandler.apxc:**

```

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');

        }

    }
}
}

```

### **MothlyEmailScheduler.apxc:**

```

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 = 'I trust this email finds you well. I am writing to remind  
you that the monthly rent is due Your timely payment ensures the smooth functioning of  
our rental arrangement and helps maintain a positive living environment for all.';  
  
    String emailSubject = 'Reminder: Monthly Rent Payment Due';  
  
    Messaging.SingleEmailMessage email = new  
    Messaging.SingleEmailMessage(); email.setToAddresses(new  
    String[] {recipientEmail}); email.setSubject(emailSubject);  
    email.setPlainTextBody(emailContent);  
  
    Messaging.sendEmail(new Messaging.SingleEmailMessage[] {email});  
}  
}  
}
```

**GitHub Link:** <https://github.com/AswiniAmaraadi/APSCHE---STB3>

**Demo Video Link:**

[https://drive.google.com/file/d/1qi7jVAObIjhRHv5Qtfw5zoMrErNX-6I/view?usp=drive\\_link](https://drive.google.com/file/d/1qi7jVAObIjhRHv5Qtfw5zoMrErNX-6I/view?usp=drive_link)