

# LEASEMANAGEMENT

College Name: Hindusthan College Of Science And Commerce College Code: CA

**TEAM ID: NM2025TMID28177 TEAM MEMBERS:**

**Team Leader Name: DHARANIDHARAN S**

**Email: [21dharani21s@gmail.com](mailto:21dharani21s@gmail.com)**

**Team Member: HARISH RAGAVENDRA S**

**Email:[harishragavendra9306@gmail.com](mailto:harishragavendra9306@gmail.com)**

**Team Member : SAKTHIVEL**

**Email: [sakthivelseeni743@gmail.com](mailto:sakthivelseeni743@gmail.com)**

**Team Member : INDHIRESH**

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

**Team Member : SANTHOSH S**

**Email: [santoshabariss513@gmail.com](mailto:santoshabariss513@gmail.com)**

## 1. INTRODUCTION

### 1. Project Overview

This project is focused on the development and implementation of a Lease Management System using Salesforce, designed to address the challenges of manual property lease tracking, tenant management, payment processing, and

communication within the organization. The goal is to deliver a comprehensive solution that streamlines and automates various aspects of lease management, from contract creation to payment tracking, approval workflows, and reporting.



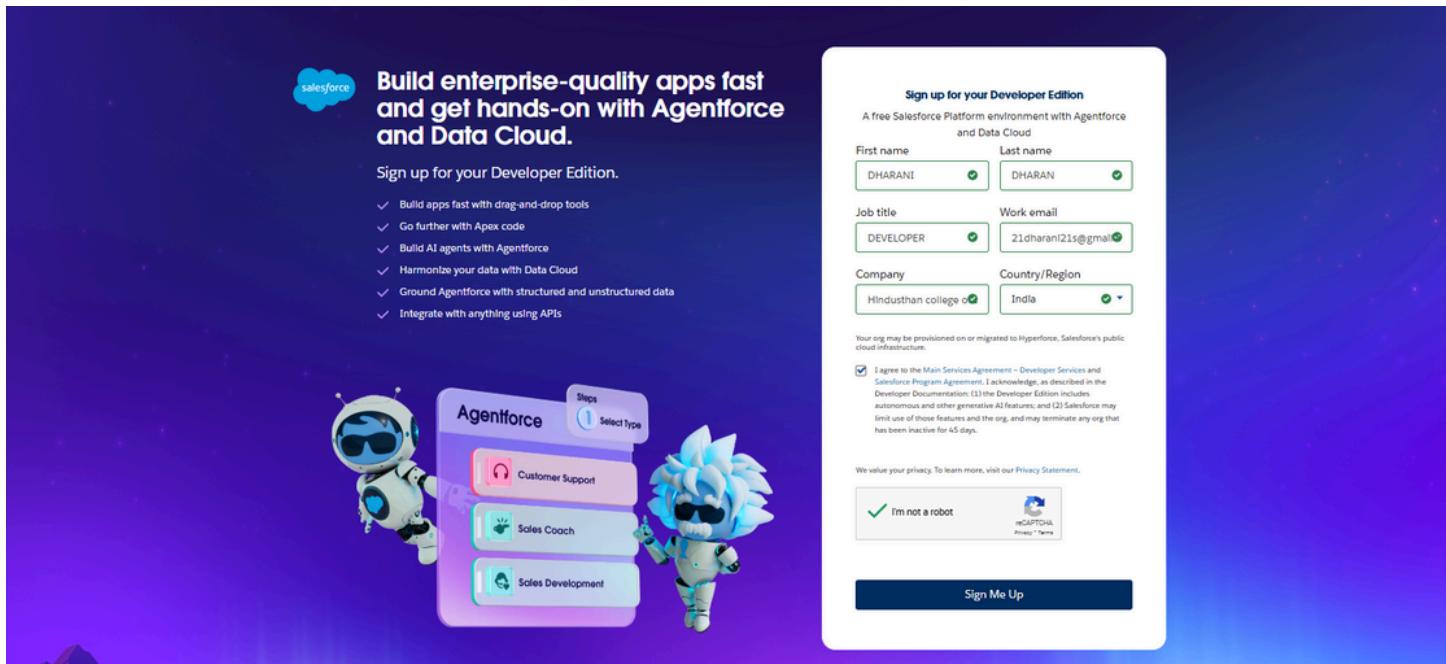
## 1. Purpose

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.

# DEVELOPMENT PHASE

**Creating Developer Account:**

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



- Created objects: Property, Tenant, Lease, Payment

A screenshot of the Salesforce Object Manager. The top navigation bar shows "SETUP > OBJECT MANAGER" and the object name "Tenant". The main area displays the "Details" tab for the Tenant object. On the left is a sidebar with links like "Field & Relationships", "Page Layouts", "Lightning Record Pages", "Buttons, Links, and Actions", "Compact Layouts", "Field Sets", "Object Limits", "Record Types", "Related Lookups", "Search Layouts", "List View Button Layout", "Restriction Rules", and "Sourcing Rules". The main "Details" pane shows the following fields:

- Description:
- API Name: Tenant\_\_c
- Custom:
- Singular Label: Tenant
- Plural Label: Tenants

On the right, there are sections for "Enable Reports" (with checkboxes for "Track Activities", "Track Feed History", and "Deployment Status"), "Deployed" (with a "Help Settings" link), and "Standard salesforce.com Help Window".

**New Custom Object**

Custom Object Definition Edit

Custom Object Information

The singular and plural labels are used in tabs, page layouts, and reports.

Label:  Example: Account

Plural Label:  Example: Accounts

Starts with vowel sound:

The Object Name is used when referencing the object via the API.

Object Name:  Example: Account

Description:

Context-sensitive Help Setting:  Open the standard Salesforce.com Help & Training window  
                                  Open a window using a Visualforce page

Content Name:

Enter Record Name Label and Format.

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is 'Account Name' and for Case it is 'Case Number'. Note that the Record Name field is always called 'Name' when referenced via the API.

Record Name:  Example: Account Name

Data Type:  Warning: If you plan to insert a high volume of records in this object, via the API for example, use the Text data type.

Optional Features

- Allow Record
- Allow New Activities
- Task Field History
- Allow in Chatter Groups
- Enable Licensing

**lease**

Details

Description

API Name:  Custom:

Singular Label:  Plural Label:

Enable Reports:  Stock Activities:  Track Field History:

Deployment Status: Deployed Help Settings: Standard salesforce.com Help Window

**Payment for tenantat**

Details

Description

API Name:  Custom:

Singular Label:  Plural Label:

Enable Reports:  Stock Activities:  Track Field History:

Deployment Status: Deployed Help Settings: Standard salesforce.com Help Window

## ■ Configured fields and relationships

Fields & Relationships				
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)		✓
sfp	sfp_c	Text(18)		
Type	Type_c	Picklist		

Fields & Relationships				
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(30)		✓

Setup Home Object Manager

SETUP > OBJECT MANAGER

## lease

Details Fields & Relationships

Fields & Relationships

	FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Page Layouts	Created By	CreatedById	Lookup(User)		
Lightning Record Pages	End date	End_date__c	Date		
Buttons, Links, and Actions	Last Modified By	LastModifiedById	Lookup(User)		
Compact Layouts	Lease Name	Name	Text(80)		✓
Field Sets	Owner	OwnerId	Lookup(User/Group)		✓
Object Limits	property	property__c	Lookup(property)		✓
Record Types	start date	start_date__c	Date		
Related Lookup Filters					
Search Layouts					
List View Button Layout					
Restriction Rules					
Scoping Rules					

Setup Home Object Manager

SETUP > OBJECT MANAGER

## Tenant

Details Fields & Relationships

Fields & Relationships

	FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Page Layouts	Created By	CreatedById	Lookup(User)		
Lightning Record Pages	Email	Email__c	Email		
Buttons, Links, and Actions	Last Modified By	LastModifiedById	Lookup(User)		
Compact Layouts	Owner	OwnerId	Lookup(User/Group)		✓
Field Sets	Phone	Phone__c	Phone		
Object Limits	status	status__c	Picklist		
Record Types	Tenant Name	Name	Text(80)		✓
Related Lookup Filters					
Search Layouts					
List View Button Layout					
Restriction Rules					
Scoping Rules					

- Developed Lightning App with relevant tabs

The screenshot shows the 'App Details & Branding' tab in the Lightning App Builder. On the left, a sidebar lists 'App Details & Branding', '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 leasing real estate properties.). It also includes sections for 'App Branding' (with an image and primary color hex value #0070C2) and 'Org Theme Options' (checkbox for using the app's image and color instead of the org's custom theme). A preview of the app launcher is shown at the bottom.

The screenshot shows the 'Navigation Items' tab in the Lightning App Builder. The sidebar includes 'App Details & Branding', 'App Options', 'Utility Items (Desktop Only)', and 'Navigation Items'. The main area displays a list of 'Available Items' on the left and 'Selected Items' on the right. The 'Available Items' list includes: Accounts, Activation Targets, Activations, All Sites, Alternative Payment Methods, Analytics, App Launcher, Appointment Categories, Appointment Invitations, Approval Requests, Case, Contact, Lead, Opportunity, Product, Record Type, Report, Task, User, and Work Order. The 'Selected Items' list on the right contains: Payment, Tenants, property, and lease.

Lightning App Builder | App Settings | Pages | Lease Management | ? Help

**User Profiles**

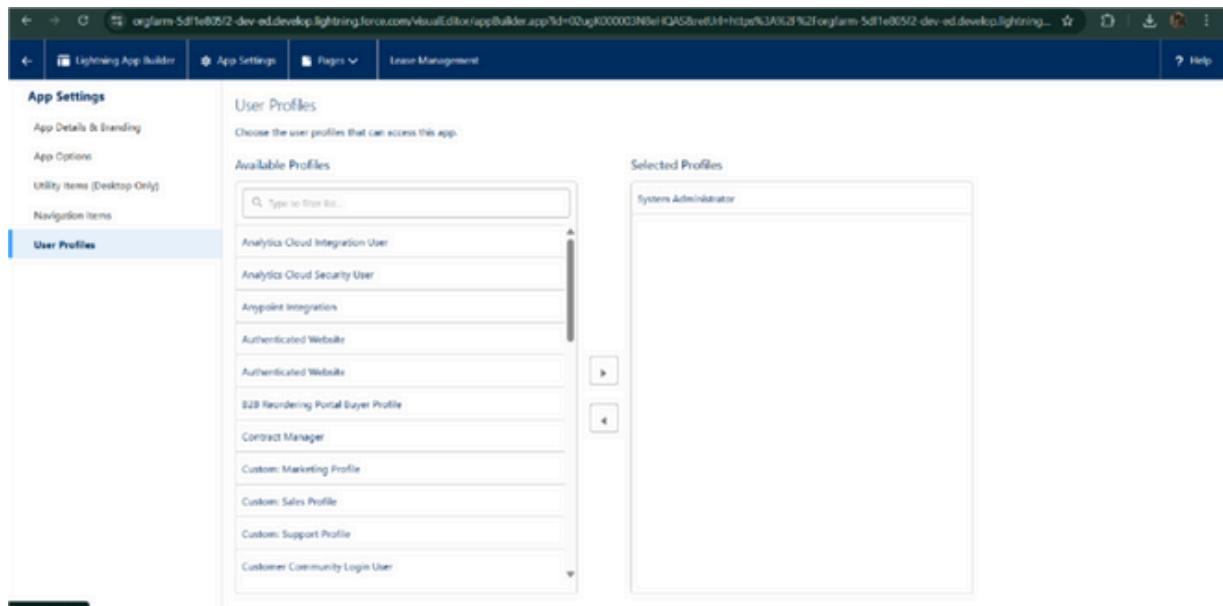
Choose the user profiles that can access this app.

**Available Profiles**

- Analytics Cloud Integration User
- Analytics Cloud Security User
- Anypoint Integration
- Authenticated Website
- Authenticated Website
- B2B Reordering Portal Buyer Profile
- Contact Manager
- Custom: Marketing Profile
- Custom: Sales Profile
- Custom: Support Profile
- Customer Community Login User

**Selected Profiles**

- System Administrator



Lease Management | Payment | Tenants | property | Lease | ?

**Payment**

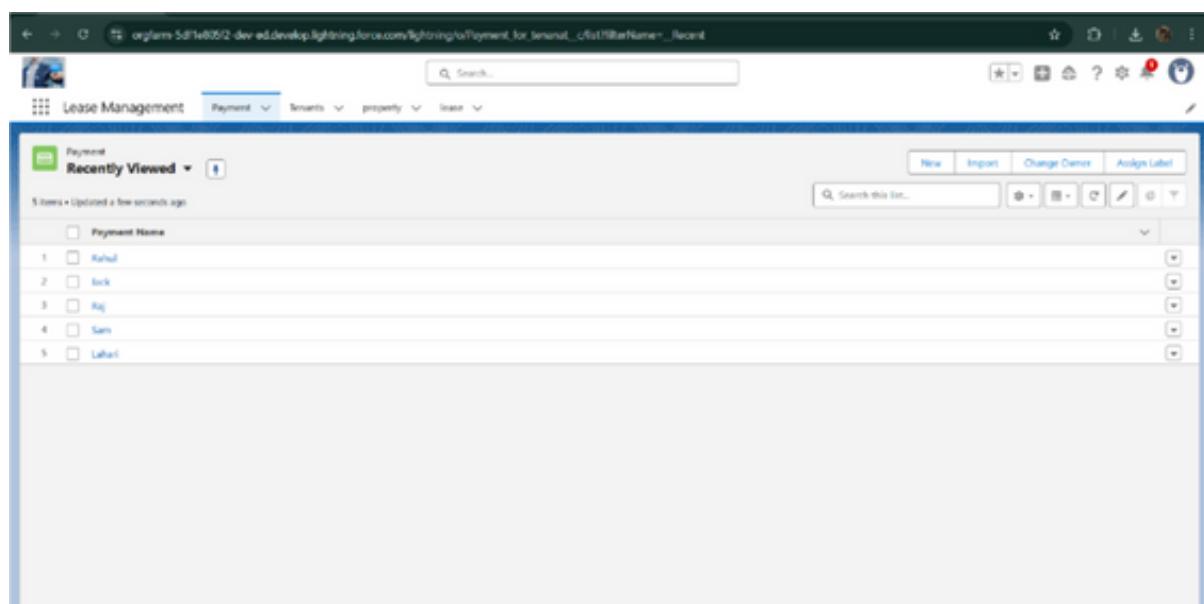
Recently Viewed

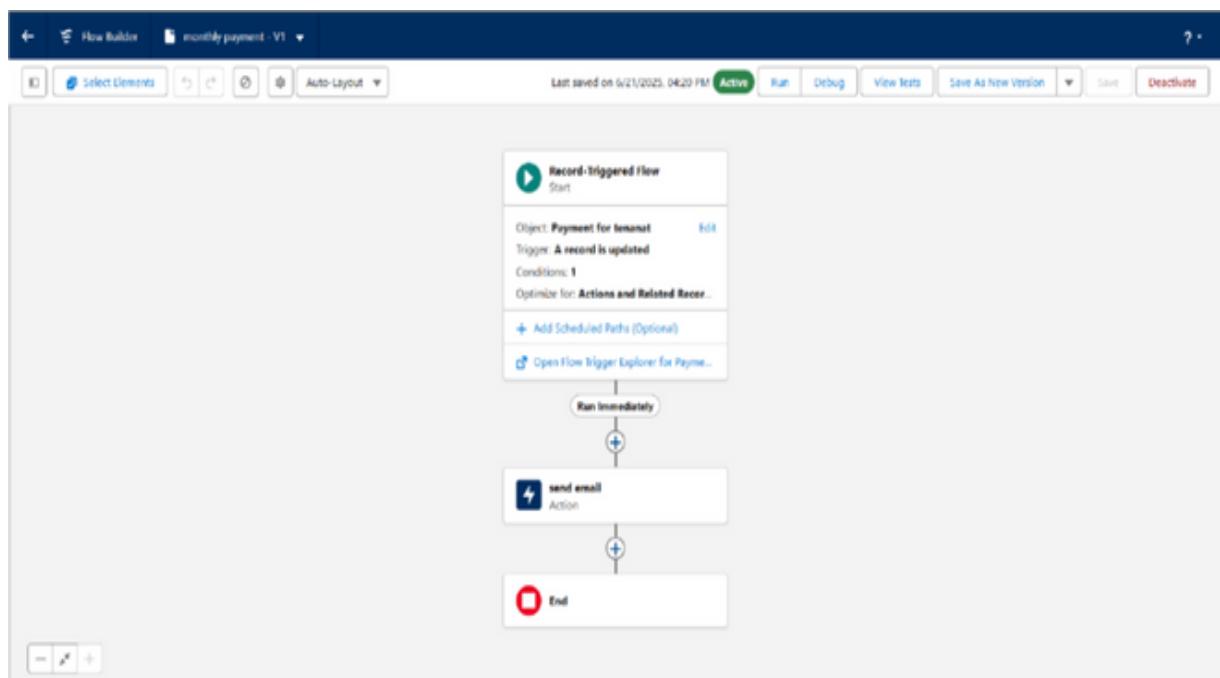
5 items • Updated a few seconds ago

	Payment Name
1	Rahul
2	Rock
3	Raj
4	Sam
5	Lakshmi

New Import Change Owner Assign Label

Search this list...



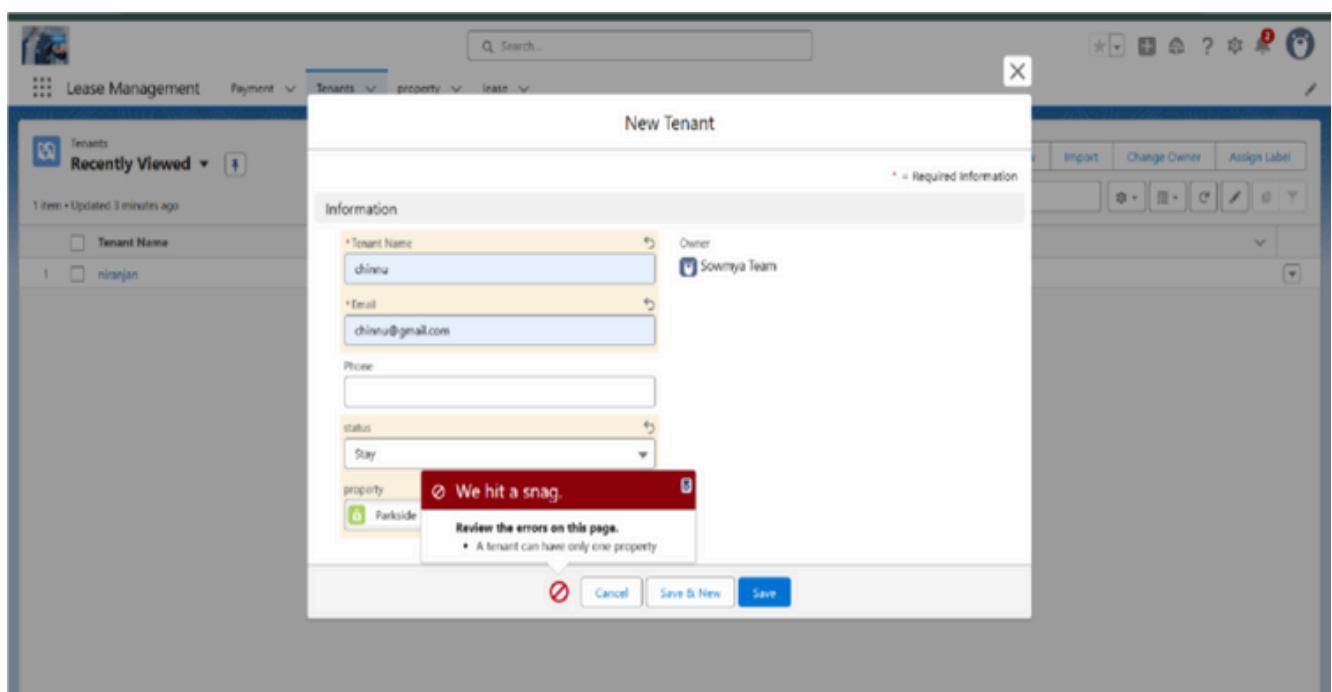


- Implemented Flows for monthly rent and payment success
- To create a validation rule to a Lease Object

The screenshot shows the Salesforce Object Manager interface for the 'lease' object. On the left, a sidebar lists various configuration options: 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 area is titled 'Validation Rule Edit' for the rule named 'lease\_end\_date'. The 'Error Condition Formula' field contains the expression 'End\_date\_\_c < start\_date\_\_c'. A dropdown menu for functions is open, showing options like ABS, ACOS, ADDMONTHS, AND, ASCII, ASIN, etc. The status bar at the bottom indicates 'Error Message'.

The screenshot shows the 'lease Validation Rule' detail page. The 'Validation Rule Detail' section displays the following information: Rule Name: 'lease\_end\_date', Error Condition Formula: 'End\_date\_\_c < start\_date\_\_c', Error Message: 'Your End-date must be greater than start-date', Description: 'Created By: Senthil.Than, 9/19/2025, 5:37 AM', Active: checked, Error Location: 'Start Date', and Modified By: 'Senthil.Than, 6/6/2026, 7:47 AM'. The sidebar on the left is identical to the previous screenshot, listing the same configuration options for the 'lease' object.

- Added Apex trigger to restrict multiple tenants per property



- Scheduled monthly reminder emails using Apex class

```

1 * global class MonthlyRentalScheduler 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    public static void sendMonthlyEmails() {
15
16        List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
17
18        for (Tenant__c tenant : tenants) {
19
20            String recipientEmail = tenant.Email__c;
21
22            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';
23
24            String emailSubject = 'Reminder: Monthly Rent Payment Due';
25
26            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
27
28            email.setToAddresses(new String[]{recipientEmail});
29
30            email.setSubject(emailSubject);
31
32            email.setPlainTextBody(emailContent);
33
34        }
35    }

```

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

**Classic Email Templates**

**Leave approved**

Preview your email template below

Email Template Detail	Actions
Email Templates from Salesforce Email Template Name: Leave approved Template Unique Name: Leave_approved Preexisting Author: Sumeet_Sabot Description: Created By: Sumeet_Sabot 6/20/2020, 1:06 AM	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Close</a>
Available For Use: <input checked="" type="checkbox"/> Last Used Date: <span style="font-size: small;">Never used</span> <a href="#">Help for this Page</a>	
<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Close</a>	

**Email Template**

[Send Test and Verify Merge Fields](#)

**Subject:** Leave approved

**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.

Q Search Setup

Setup Home Object Manager

Q\_email template

v Email

Classic Email Templates Lightning Email Templates

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

Test Email Template **Tenant leaving**

Preview your email template below.

Email Template Detail

Email Templates from Salesforce	Email Template Name: Tenant leaving	Available For Use: <input checked="" type="checkbox"/>
Template Unique Name:	tenant_leaving	Last Used Date:
Encoding:	Unicode (UTF-8)	Times Used:
Author:	SOMETHING_SFDC	
Description:	SOMETHING_SFDC	
Created By:	SOMETHING_SFDC, 6/09/2025, 1:06 AM	Modified By: SOMETHING_SFDC, 6/09/2025, 1:06 AM

Email Template

Send Test and Verify Merge Fields

Subject: request for approve the leave

Plain Text Preview

Dear [Tenant\_\_c.CreationDate],  
Please approve my leave.

Q Search Setup

Setup Home Object Manager

Q\_email template

v Email

Classic Email Templates Lightning Email Templates

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

Test Email Template **Leave rejected**

Preview your email template below.

Email Template Detail

Email Templates from Salesforce	Email Template Name: Leave rejected	Available For Use: <input checked="" type="checkbox"/>
Template Unique Name:	Leave_rejected	Last Used Date:
Encoding:	Unicode (UTF-8)	Times Used:
Author:	SOMETHING_SFDC	
Description:	SOMETHING_SFDC	
Created By:	SOMETHING_SFDC, 6/09/2025, 1:11 AM	Modified By: SOMETHING_SFDC, 6/09/2025, 1:11 AM

Email Template

Send Test and Verify Merge Fields

Subject: Leave rejected

Plain Text Preview

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:

**Classic Email Templates**

**Tenant Email**

Email Template Detail

Email Templates from Salesforce		Unified Public Classic Email Templates	
Email Template Name	Tenant Email	Available For Use	<input checked="" type="checkbox"/>
Template Unique Name	Tenant_Email	Last Used Date	
Encoding	Unicode (UTF-8)	Times Used	
Author	Sistema, Team (Chatter)		
Description			
Created By	Sistema, Team (6/20/2025, 1:12 AM)	Modified By	Sistema, Team (6/20/2025, 1:12 AM)

**Email Template**

Subject: Urgent: Monthly Rent Payment Reminder

Plain Text Preview:

```
Dear {Tenant__c.Name},  
  
I hope this email finds you well. We appreciate your continued tenancy of our property and I hope you have been comfortable in your residence.
```

This template is used in a Tenant account. It has been used once. Last modified: 6/20/2025, 1:12 AM by Sistema, Team.

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

**Classic Email Templates**

**Tenant payment**

Email Template Detail

Email Templates from Salesforce		Unified Public Classic Email Templates	
Email Template Name	Tenant_payment	Available For Use	<input checked="" type="checkbox"/>
Template Unique Name	Tenant_payment	Last Used Date	
Encoding	Unicode (UTF-8)	Times Used	
Author	Sistema, Team (Chatter)		
Description			
Created By	Sistema, Team (6/20/2025, 1:13 AM)	Modified By	Sistema, Team (6/20/2025, 1:13 AM)

**Email Template**

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:

The screenshot shows the 'Approval Processes' page in the Salesforce Setup. The process is named 'Tenant: TenantApproval'. It has one step, 'Step 1', which is a 'Record Lock' action. The process is active and was created by 'Screena Team' on 6/25/2025 at 3:01 AM.

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

For Check for Vacant:

The screenshot shows the 'Approval Processes' page in the Salesforce Setup. The process is named 'Tenant: check for vacant'. It has two steps: 'Record Lock' and 'Email Alert'. The process is active and was created by 'Screena Team' on 6/25/2025 at 3:18 AM.

Action	Type	Description
Record Lock	Lock the record from being edited	
Email Alert	DEBBD_ABDSVVE.MV.JOBJS	

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

- Apex Trigger

## Create an Apex Trigger

The screenshot shows the Salesforce IDE interface. The top navigation bar includes File, Edit, Debug, Test, Workspace, Help, and a dropdown for Tooling API Version. The main area displays the code for a trigger:

```
1 trigger test on Tenant__c (before insert)
2
3 {
4     if(trigger.isInsert && trigger.isBefore){
5         testHandler.preventInsert(trigger.new);
6     }
7 }
```

A modal dialog titled "Open" is displayed, listing various types of objects in the repository:

Entity Name	Edition	Reflected
Body Type	Standard	
Classes		
Triggers	test	
Pages		
Page Components		
Objects		
Static Resources		
Packages		

At the bottom of the dialog are buttons for Open, Filter, Refresh, and a note about Managed Packages.

The bottom navigation bar includes tabs for Logs, Tests, Checkstyle, Query Editor, View Status, Progress, Problems, and a dropdown for "None". Below the tabs are buttons for Line and Problem.

Developer Console - Google Chrome  
cegfarm-5d1fa0502-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSPage

File • Edit • Debug • Test • Workspace • Help •

test.apex | testHandler.apex | **PropertyEmailScheduler.apex** |

Code Coverage: None | API Version: 54.0 | Go To

```
1 trigger test on Tenant__c (before insert)
2
3 {
4     if(trigger.isInsert && trigger.isBefore){
5         testHandler.preventInsert(trigger.new);
6     }
7 }
8
9 }
```

Logs Tests Checkpoints Query Editor View Status Progress Problems

None

Create an Apex Handler class

Developer Console - Google Chrome  
cegfarm-5d1fa0502-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSPage

File • Edit • Debug • Test • Workspace • Help •

test.apex | **testHandler.apex** | PropertyEmailScheduler.apex |

Code Coverage: None | API Version: 54.0 | 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.Id);
10    }
11
12    }
13
14    for (Tenant__c newTenant : newList) {
15
16        if (newTenant.Property__c != null) {
17
18            newTenant.addError('A tenant must have a property');
19        }
20    }
21
22 }
```

Entity Name: Existing Tenant | Edit Schema | Related

Entity Name	Relationship	Related
Existing Tenant	Name	ApexTrigger
Existing Tenant	IsDeleted	CustomField
Existing Tenant	Properties	Object
Existing Tenant	Parent Components	References
Existing Tenant	Schema	References
Existing Tenant	Shared Permissions	References
Existing Tenant	Package	References

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

Logs Tests Checkpoints Query Editor View Status Progress Problems

None

Developer Console - Google Chrome

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

Toolbars: testHandler.apex HeadlessBatchScheduler.apex

Code Coverage: None • API Version: 54 • Go To

```

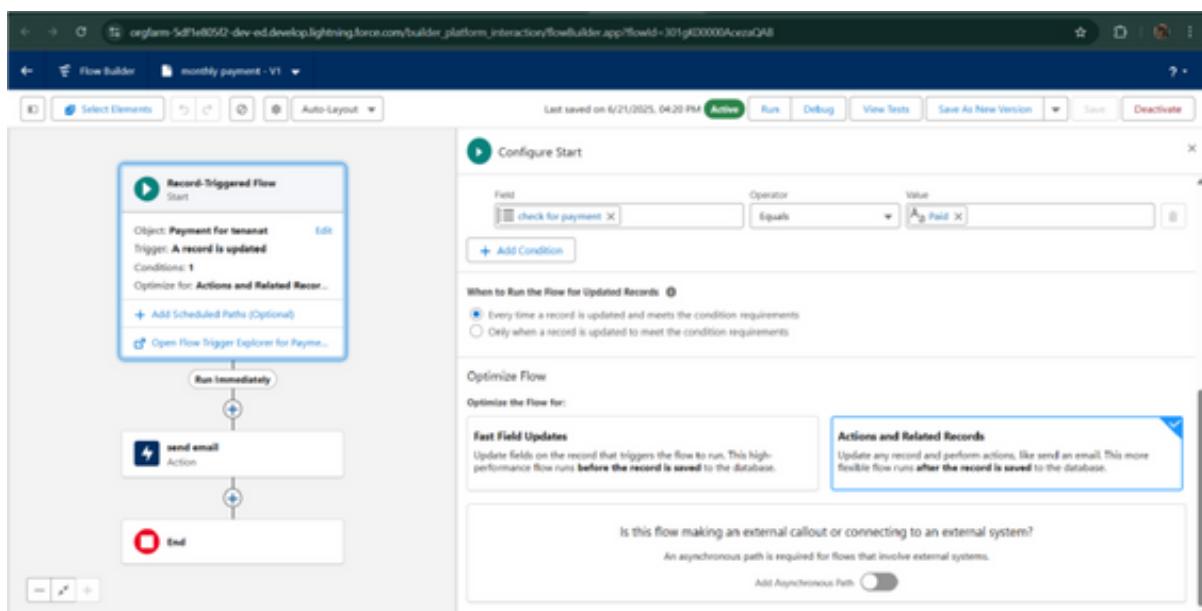
1 * public class testHandler {
2
3     public static void preventInsert(List<Tenant__c> newList) {
4
5         Set<Id> existingPropertyIds = new Set<Id>();
6
7         for (Tenant__c existingTenant : [SELECT Id, Property__c FROM Tenant__c WHERE Property__c != null]) {
8
9             existingPropertyIds.add(existingTenant.Property__c);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

```

Logs Test Checkpoints Query Editor View Status Progress Problems

None Line Problem

- FLOWS



The screenshot shows the Flow Builder interface with the following details:

- Flow Name:** monthly payment - V1
- Last saved on:** 6/21/2025, 04:20 PM
- Status:** Active
- Object:** Payment for tenant
- Trigger:** A record is updated
- Conditions:** 1
- Optimize for:** Actions and Related Records
- Run Immediately:** Enabled
- Actions:**
  - send email
  - End

**Configure Start**

**Select Object**  
Select the object whose records trigger the flow when they're created, updated, or deleted.

**Object:** Payment for tenant

**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**  
All Conditions Are Met (AND)

- Schedule class: Create an Apex Class

The screenshot shows the Salesforce Developer Console interface. The top navigation bar includes 'Developer Console', 'Google Chrome', and the URL 'codeforce-S0f1e050-dev-ed-develop.my.salesforce.com/s/common/open/debug/ApexCodePage'. The menu bar has items like File, Edit, Debug, Test, Workspaces, Help, and a 'Tool Input' dropdown. A 'Code Coverage' tab is selected, showing 'Name = MonthlyEmailScheduler.apex' and 'API Version = 54'. The main code editor area contains the following Apex code:

```
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             String recipientEmail = tenant.Email__c;
22
23
24
25
26
27
28
29
29
```

A modal dialog titled 'Open' is displayed over the code editor, listing objects in the repository. The 'Entity Type' dropdown is set to 'Object'. The 'Selected' row is 'MonthlyEmailScheduler'. Other listed entities include 'Email', 'CustomObj...', 'Referenced', 'Email', 'CustomObj...', 'Referenced', and 'Email', 'CustomObj...', 'Referenced'. The bottom of the dialog has 'Open', 'Edit', 'Filter', 'Hide Managed Packages', and 'Refresh' buttons.

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is `oxygen-Sf1fe0502-dev-ed.develop.my.salesforce.com/sf/common/apex/debug/ApexCSPage`. The page displays an Apex class named `RentalReminder` with a single method `sendMonthlyRental()`. The code uses the `ApexEmailMessage` class to construct an email message, setting the subject to "Monthly Rent Remind" and the body to a friendly message about timely payment. It then sends the email to the `tenant_email` field of the `Rental` object. The developer console interface includes tabs for Home, Developers, My Setup, Help, and Logout.

```
1 // apex:page
2 // apex:form
3
4 public class RentalReminder {
5
6     public void sendMonthlyRental() {
7
8         Integer currentday = Date.Today().Day();
9
10        if (currentday == 1) {
11            sendEmail();
12        }
13    }
14
15
16    public void sendEmail() {
17
18        List<Rental> tenanted = [SELECT id, tenant_email FROM Rental];
19
20        for (Rental__c tenant : tenanted) {
21
22            String recipientEmail = tenant.tenant_email;
23
24            String rentalId = '12345678901234567890123456789012';
25
26            Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
27
28            mail.setTo(recipientEmail);
29            mail.setSubject('Monthly Rent Remind');
30            mail.setPlainText('Hello ' + tenant.tenant_name + ',');
31
32            mail.setHTMLBody('
33                RENT DUE!
34                Your monthly rent is due now. Please make sure to pay it on time to ensure the smooth functioning of our rental arrangement and help maintain a positive living environment for all.
35            ');
36
37            Messaging.sendEmail(new Messaging.SingleEmailMessage[] {mail });
38        }
39    }
40 }
```

## Schedule Apex class

Setup Home Object Manager

Q apex

Apex Classes

Apex Class Detail

Name: MonthlyEmailScheduler

Namespace Prefix: Created By: Somaia.Zeab, 6/23/2025, 2:46 AM

Status: Active Code Coverage: 0% (0/15)

Last Modified By: Somaia.Zeab, 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) {  
            sendMonthlyEmail();  
        }  
    }  
  
    public static void sendMonthlyEmail() {  
        List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];  
        for (Tenant__c tenant : tenants) {  
    }  
}
```

Lease Management Payment Tenants Properties lease

Tenant Janani

Related Details

Tenant Name: Janani

Email: aswinjanani2011@gmail.com

Phone:

status: Stay

Property: House

Created By: JANANI P, 9/5/2025, 5:16 AM

Last Modified By: JANANI P, 9/8/2025, 7:54 AM

Owner: JANANI P

Activity

Filters: All time • All activities • All types

Upcoming & Overdue

No activities to show.

Get started by sending an email, scheduling a task, and more.

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

The screenshot shows a Microsoft Dynamics 365 Lease Management interface. In the top navigation bar, there are links for Payment, Tenants, Properties, and Lease. A search bar is located at the top center. On the left, a sidebar titled 'Lease Management' shows a tree view with 'Tenants' selected, and a sub-tree for 'Janani'. The main content area displays a 'Details' tab for a tenant named 'Janani'. The details include fields for Tenant Name, Email, Phone, status, Property, and Created By. The 'Owner' field is populated with 'JANANI P'. The 'Last Modified By' field shows 'JANANI P' with a timestamp of '9/8/2025, 7:54 AM'. On the right side, a 'Notifications' pane is open, showing a list of five notifications all from 'Janani' with the subject 'Approval request for the tenant is approved'. The notifications are dated Sep 8, 2025, at various times: 8:24 PM, 8:21 PM, 2:42 PM, and 2:39 PM.

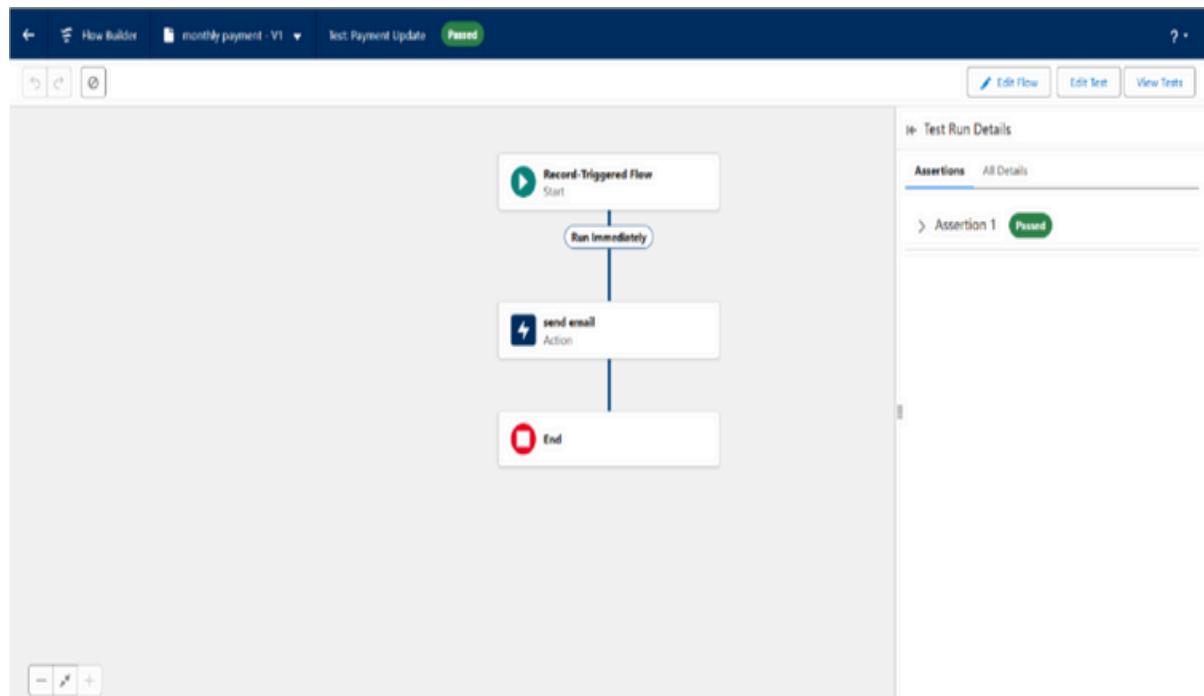
# FUNCTIONAL AND PERFORMANCE TESTING

## Performance Testing

- Trigger validation by entering duplicate tenant-property records



- Validation Rule checking



Lease Management

Payment ▾

Tenants ▾

Properties ▾

lease ▾

Search...

Notifications

Mark all as read

Approval request for the tenant is approved  
Janani  
3 hours ago

Approval request for the tenant is approved  
Janani  
Sep 8, 2025, 8:24 PM

Approval request for the tenant is approved  
Janani  
Sep 8, 2025, 8:21 PM

Approval request for the tenant is approved  
Janani  
Sep 6, 2025, 2:42 PM

Approval request for the tenant is approved  
Janani  
Sep 6, 2025, 2:39 PM

Tenant  
Janani

Related Details

Tenant Name  
Janani

Email  
aswinjanani2011@gmail.com

Phone

status  
Stay

Property  
House

Created By  
JANANI P., 9/5/2025, 5:16 AM

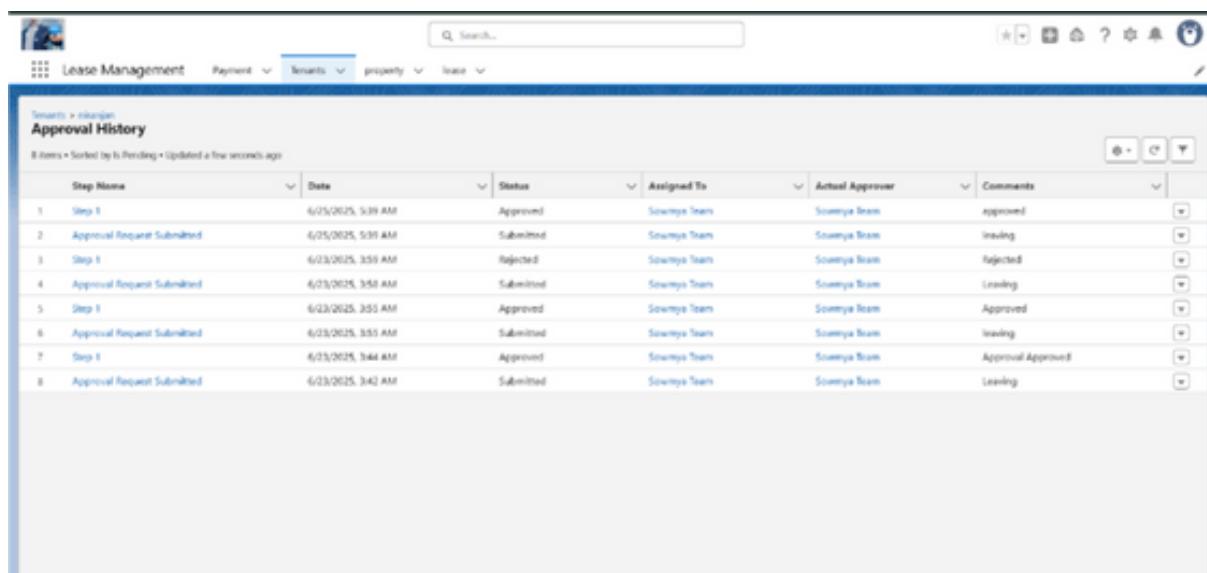
Last Modified By  
JANANI P., 9/8/2025, 7:54 AM

- Test flows on payment update

# RESULTS

Action	Label	Tab Style	Description
Link   Get	Basic		
Link   Get	Element		
Link   Get	Search		
Link   Get	Results		

# Output Screenshots



The screenshot shows a software interface titled "Lease Management". The top navigation bar includes links for "Payment", "Tenants", "property", and "Leave". A search bar is located at the top right. Below the navigation, a breadcrumb trail reads "Tenants > [unclear] Approval History". A message indicates "8 items • Sorted by Is Pending • Updated a few seconds ago". The main content is a table titled "Approval History" with the following columns: Step Name, Date, Status, Assigned To, Actual Approver, and Comments. The table contains 8 rows of data.

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	6/25/2025, 5:51 AM	Rejected	Sowmya Team	Sowmya Team	Rejected
4 Approval Request Submitted	6/25/2025, 5:51 AM	Submitted	Sowmya Team	Sowmya Team	Leaving
5 Step 1	6/25/2025, 5:55 AM	Approved	Sowmya Team	Sowmya Team	Approved
6 Approval Request Submitted	6/25/2025, 5:55 AM	Submitted	Sowmya Team	Sowmya Team	leaving
7 Step 1	6/25/2025, 5:55 AM	Approved	Sowmya Team	Sowmya Team	Approval Approved
8 Approval Request Submitted	6/25/2025, 5:55 AM	Submitted	Sowmya Team	Sowmya Team	Leaving

- Request for approve the leave

Approval Processes | Sales X Approval Processes | Sales X SmartSearch X Approval Approval | Sales X Latha | Team | Salesforce X request for approve the leave X +

mail.google.com/mail/u/0/?tab=rmlogblfinbox/FMfcgjQh/dz/TNwvNGGJDNgJ.PVIZNm

Gmail Search mail

Compose

Inbox

Stared

Snoozed

Sent

Drafts

More

Labels

request for approve the leave

Sowmya Team sowmyavale@gmail.com 11:36 AM (4 minutes ago)

to me

Dear Sowmya Team,

Please approve my leave

Sowmya Team sowmyavale@gmail.com 11:39 AM (3 minutes ago)

to me

\*\*\*

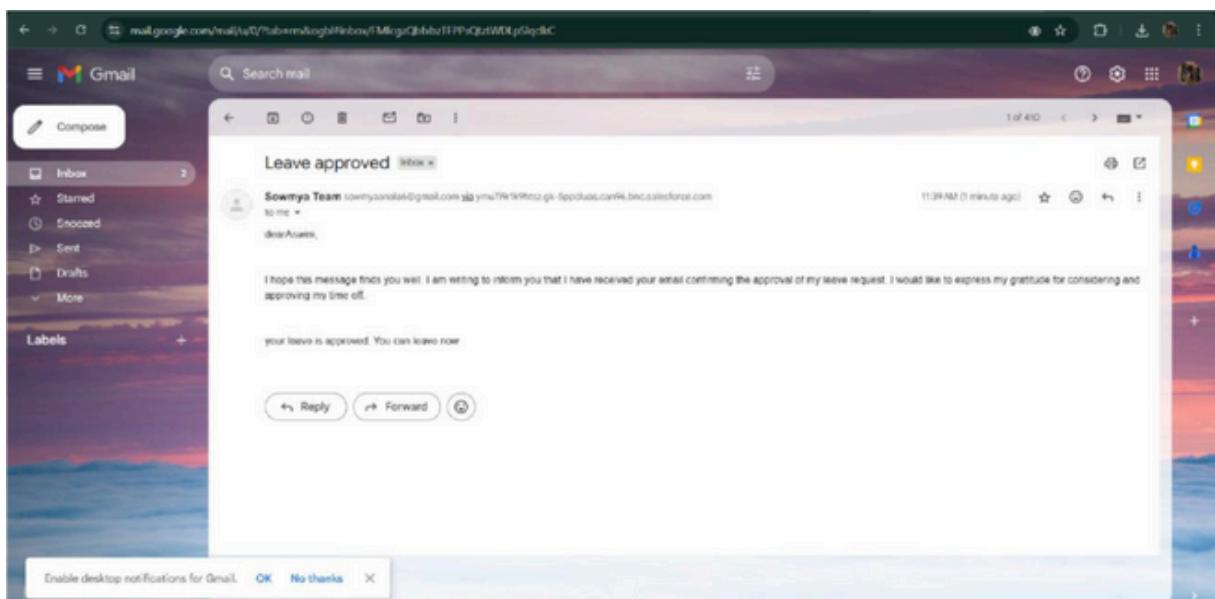
Reply Forward

Enable desktop notifications for Gmail. OK No thanks

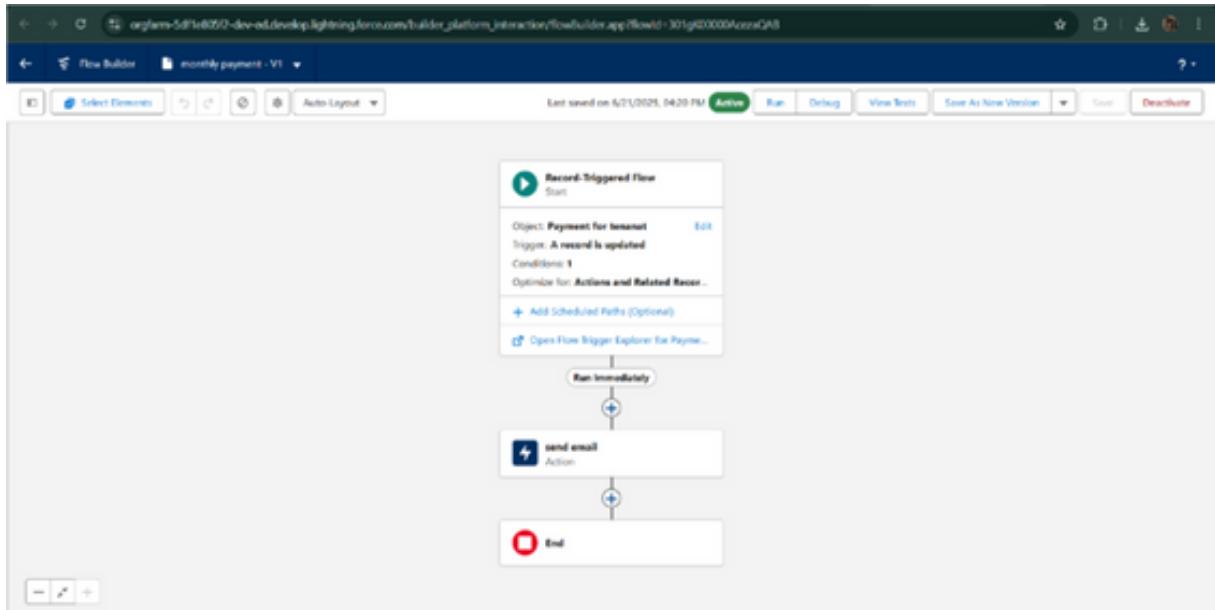
The screenshot shows a Gmail inbox with a sunset background. There are two messages in the list:

- request for approve the leave** (11:36 AM, 4 minutes ago)  
Sowmya Team sowmyavale@gmail.com  
to me  
Dear Sowmya Team,  
Please approve my leave
- Sowmya Team** sowmyavale@gmail.com 11:39 AM (3 minutes ago)  
to me  
\*\*\*

At the bottom of the screen, there is a notification bar with the text "Enable desktop notifications for Gmail." followed by "OK" and "No thanks" buttons.



## ■ Leave rejected



mail.google.com/mail/u/0/?tab=rmologbflr&cosy=MicgjQMsx2TQfHcJN1CCwGzggfDr

Gmail Search mail

Compose

Inbox

Starred

Snoozed

Sent

Drafts

More

Labels

Leave rejected

Sowmya Team <sowmyasalai@gmail.com> [to me](#)

11:45 AM (10 minutes ago)

Dear Aravind,

I hope this email finds you well. Your contract has not ended. So we can't approve your leave.

Your leave has rejected

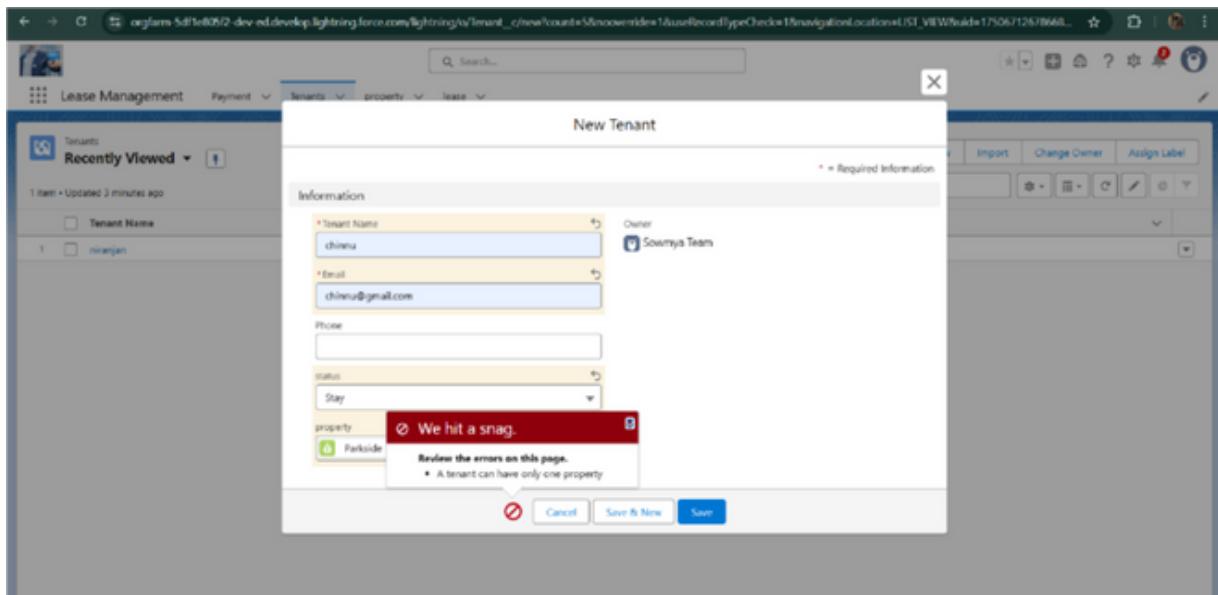
Sowmya Team <sowmyasalai@gmail.com> [to me](#)

11:45 AM (10 minutes ago)

Enable desktop notifications for Gmail. [OK](#) [No thanks](#)

The screenshot shows a Gmail inbox with a sunset background. There are two emails in the inbox. The top email is from 'Sowmya Team' with the subject 'Leave rejected'. The body of the email says: 'I hope this email finds you well. Your contract has not ended. So we can't approve your leave.' Below this, it says 'Your leave has rejected'. The second email from the same sender has the same subject 'Leave rejected' and a similar message. The inbox sidebar includes labels for 'Inbox', 'Starred', 'Snoozed', 'Sent', 'Drafts', and 'More'. A notification at the bottom asks if the user wants to enable desktop notifications for Gmail, with 'OK' and 'No thanks' buttons.

- Trigger error messages



- Approval process notifications

## ADVANTAGE

**Lower Upfront Costs:** Leasing eliminates large capital expenditures, allowing businesses to preserve cash for other needs.

**Flexibility:** Businesses can easily upgrade assets at the end of a lease term helping to avoid obsolescence.

**Tax Benefits:** Lease payments are often deductible as operating expenses, providing a tax advantage for the lessee.

# DISADVANTAGE

**Higher Long-Term Costs:** While initial costs are lower, the total cost of leasing can be higher than buying over the asset's lifespan.

**No Ownership Equity:** At the end of the lease, lessees do not gain ownership of the asset, and their payments do not build equity.

**Restrictions:** Leased assets often come with limitations, such as mileage caps for vehicles or restrictions on customization.

# CONCLUSION

## Summary of Achievements

The Salesforce implementation for Lease Management has successfully streamlined and automated critical aspects of the leasing lifecycle. Key accomplishments include:

### 1. Comprehensive Lease Lifecycle Management

- Implemented a robust system to manage lease agreements, from creation and amendments to renewals and terminations.

### 1. Automated Payment Processes

- Enabled scheduled rent tracking, payment reminders, and overdue alerts, reducing manual oversight and improving cash flow management.

## 1. Efficient Expiration and Renewal Workflows

- Automated notifications and streamlined renewal processes, ensuring timely follow-ups and improved lease retention rates.

## 1. Enhanced Property and Tenant Management

- Centralized information for properties, units, and tenants, improving operational visibility and collaboration.

## 1. Actionable Insights and Reporting

- Delivered custom reports and dashboards to monitor revenue, lease performance, and occupancy metrics, supporting data-driven decision-making.

## 1. Improved User Experience

- Provided intuitive interfaces and mobile accessibility for leasing agents, property managers, and tenants, enhancing productivity and satisfaction.

## 1. Secure and Scalable Solution

- Ensured role-based access, field-level security, and compliance with legal requirements, enabling a reliable and scalable system for lease management.

## 1. Integration with External Systems

- Established seamless connections with payment gateways and accounting systems for end-to-end process automation.

# REFERENCE

1. International Journal of Advanced Research in Science, Communication and Technology (IJARSCT):  
<https://ijarsct.co.in/Paper7646.pdf>
1. Lease-Management-System-on-Salesforce: <https://github.com/VedantRajGaur/Lease-Management-System-on-Salesforce>.
1. Github lease Management: <https://github.com/mukunthan893/NMLEASEMANAGEMENT>

1. Salesforce documentation:

Build an Integrated Real Estate CRM for Tenant Management.

## APPENDIX

- o **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});
```

```
}
```

```
}
```

```
}
```