

# LEASE MANAGEMENT

**College Name:** KAMALAM COLLEGE OF ARTS AND SCIENCE

**College Code:** Bru5s

## **TEAM ID:**

## **TEAM MEMBERS:**

**Team LeaderName:** HARIKRASANTH S

**Email:** hari18658@gmail.com

**Team Member1:** ARUNKUMAR K

**Email:** arun748468@gmail.com

**Team Member:** ABINAYA T

**Email:** abinayat069@gmail.com

**Team Member:** NISHANA F

**Email:** nishananisha181923@gmail

## 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.



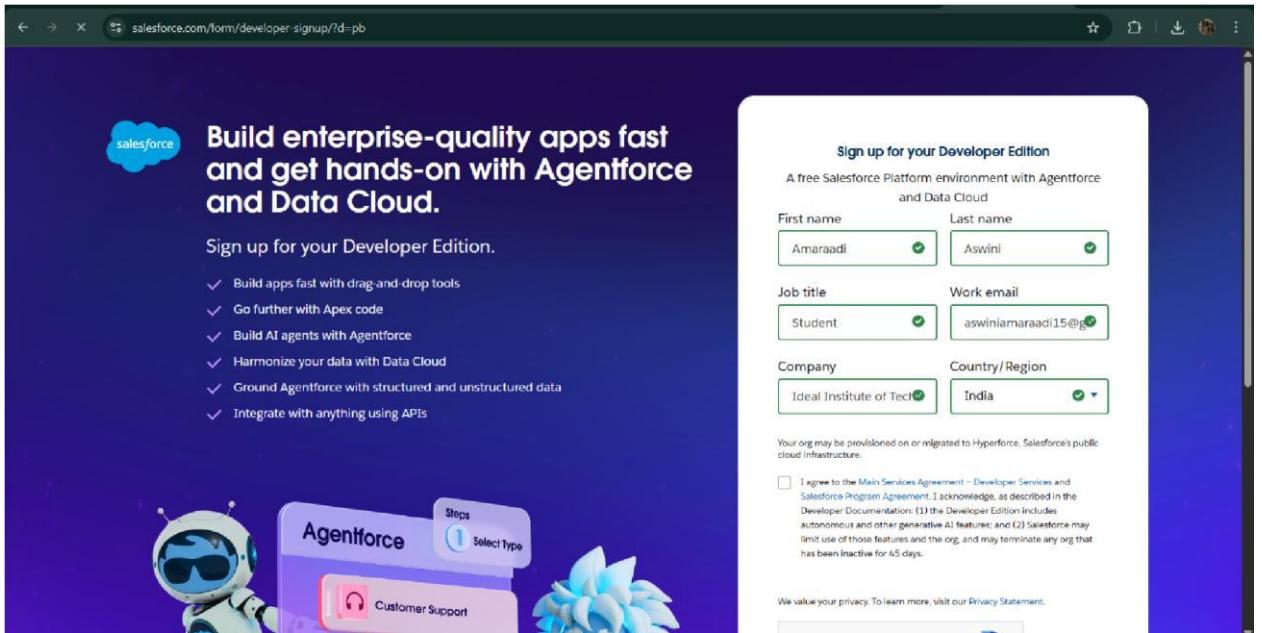
## 1.2 Purpose

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.

# DEVELOPMENT PHASE

## Creating Developer Account:

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



- Created objects: Property, Tenant, Lease, Payment

SETUP > OBJECT MANAGER

## Tenant

Details

Description

API Name  
Tenant\_c

Custom

Singular Label  
Tenant

Plural Label  
Tenants

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

Details

Enable Reports  
✓

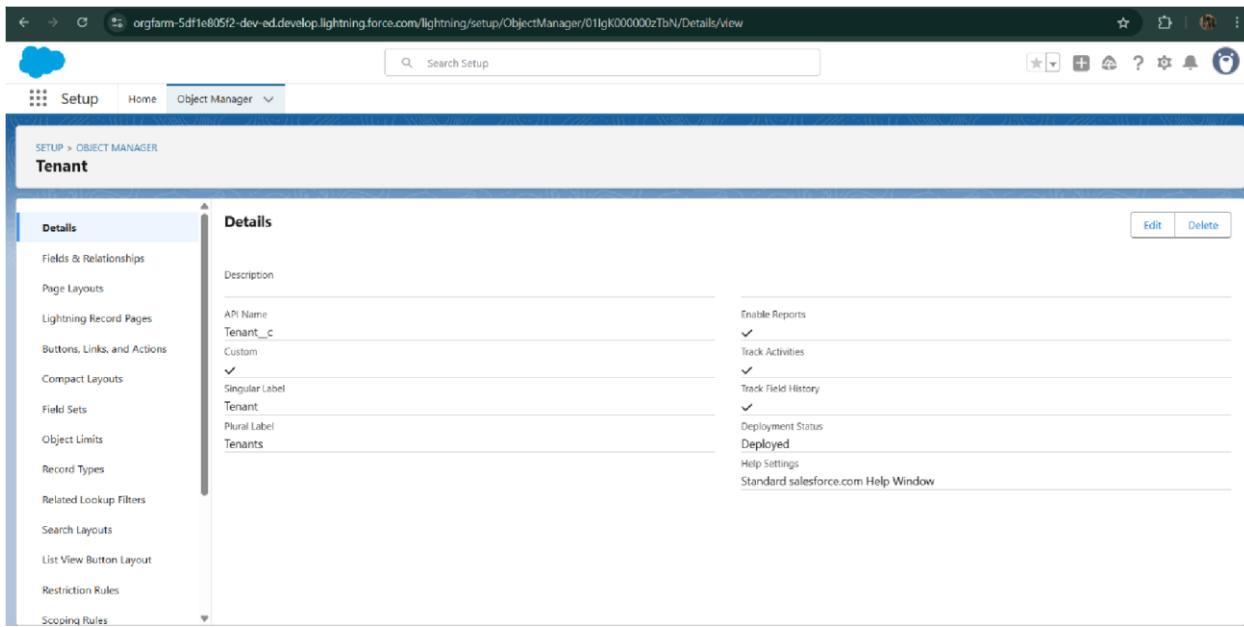
Track Activities  
✓

Track Field History  
✓

Deployment Status  
Deployed

Help Settings  
Standard salesforce.com Help Window

Edit Delete



SETUP > OBJECT MANAGER

## lease

Details

Description

API Name  
lease\_c

Custom

Singular Label  
lease

Plural Label  
lease

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

Details

Enable Reports  
✓

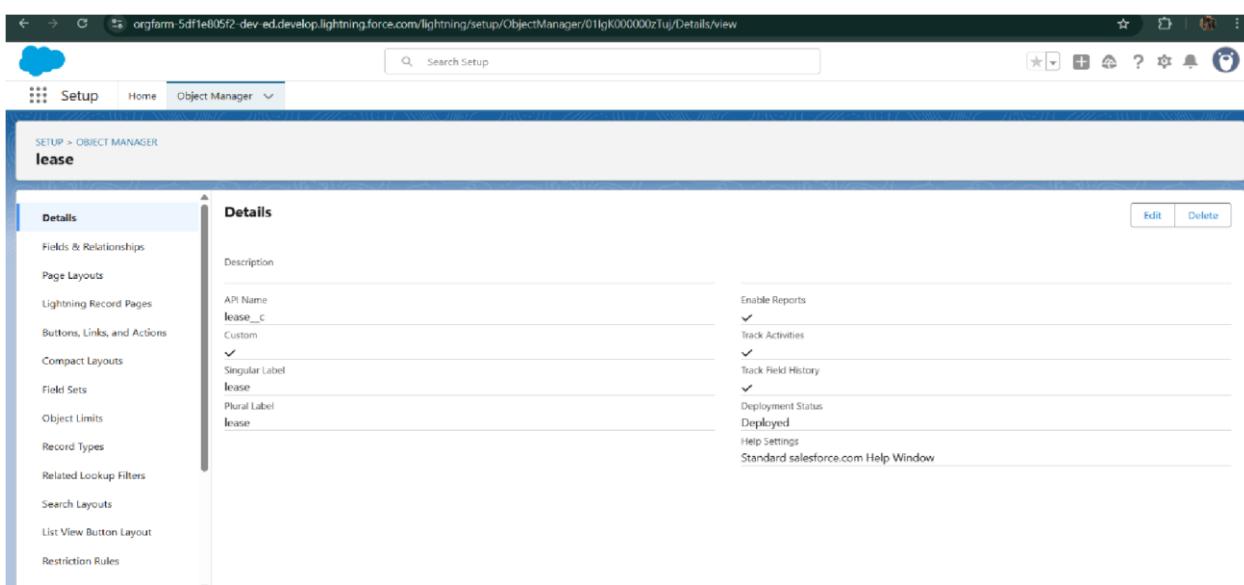
Track Activities  
✓

Track Field History  
✓

Deployment Status  
Deployed

Help Settings  
Standard salesforce.com Help Window

Edit Delete



The screenshot shows the Salesforce Setup interface under the Object Manager. A sidebar on the left lists various setup categories like Fields & Relationships, Page Layouts, and Record Types. The main pane displays the 'Details' tab for the 'Payment for tenant' object. It includes fields for API Name (Payment\_for\_tenant\_\_c), Singular Label (Payment for tenant), Plural Label (Payment), and several checkboxes for reporting and tracking features.

- Configured fields and relationships

The screenshot shows the Salesforce Setup interface under the Object Manager. The sidebar lists various setup categories. The main pane shows the 'Fields & Relationships' section for the 'property' object, listing nine fields with their corresponding field labels, field names, data types, controlling fields, and indexing status.

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(80)		
sfqt	sfqt__c	Text(18)		
Type	Type__c	Picklist		

Setup > Object Manager

### Payment for tenantat

Fields & Relationships				
7 Items, Sorted by Field Label				
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Amount	Amount_c	Number(18, 0)		<input type="checkbox"/>
check for payment	check_for_payment_c	Picklist		<input type="checkbox"/>
Created By	CreatedById	Lookup(User)		<input type="checkbox"/>
Last Modified By	LastModifiedById	Lookup(User)		<input type="checkbox"/>
Owner	OwnerId	Lookup(User,Group)		<input checked="" type="checkbox"/>
Payment date	Payment_date_c	Date		<input type="checkbox"/>
Payment Name	Name	Text(80)		<input checked="" type="checkbox"/>

Setup > Object Manager

### lease

Fields & Relationships				
7 Items, Sorted by Field Label				
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		<input type="checkbox"/>
End date	End_date_c	Date		<input type="checkbox"/>
Last Modified By	LastModifiedById	Lookup(User)		<input type="checkbox"/>
lease Name	Name	Text(80)		<input checked="" type="checkbox"/>
Owner	OwnerId	Lookup(User,Group)		<input checked="" type="checkbox"/>
property	property_c	Lookup(property)		<input checked="" type="checkbox"/>
start date	start_date_c	Date		<input type="checkbox"/>

Fields & Relationships				
7 items: Sorted by Field Label				
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Email	Email__c	Email		
Last Modified By	LastModifiedById	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

**App Details & Branding**

Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.

**App Details**

\*App Name: Lease Management

\*Developer Name: Lease\_Management

Description: Application to efficiently handle the processes related to leasing real estate properties.

**App Branding**

Image: Placeholder image

Primary Color Hex Value: #0070D2

Org Theme Options:  Use the app's image and color instead of the org's custom theme

**App Launcher Preview**

Lease Management  
Application to efficiently handle the processes relate...

[Lightning App Builder](#) [App Settings](#) [Pages](#) [Lease Management](#)

**App Settings**

App Details & Branding  
App Options  
Utility Items (Desktop Only)  
**Navigation Items**  
User Profiles

**Navigation Items**

Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.

**Available Items**

Type to filter list... [Create](#)

- Accounts
- Activation Targets
- Activations
- All Sites
- Alternative Payment Methods
- Analytics
- App Launcher
- Appointment Categories
- Appointment Invitations
- Approval Requests

**Selected Items**

- Payment
- Tenants
- property
- lease

[Lightning App Builder](#) [App Settings](#) [Pages](#) [Lease Management](#)

**App Settings**

App Details & Branding  
App Options  
Utility Items (Desktop Only)  
**User Profiles**  
Navigation Items

**User Profiles**

Choose the user profiles that can access this app.

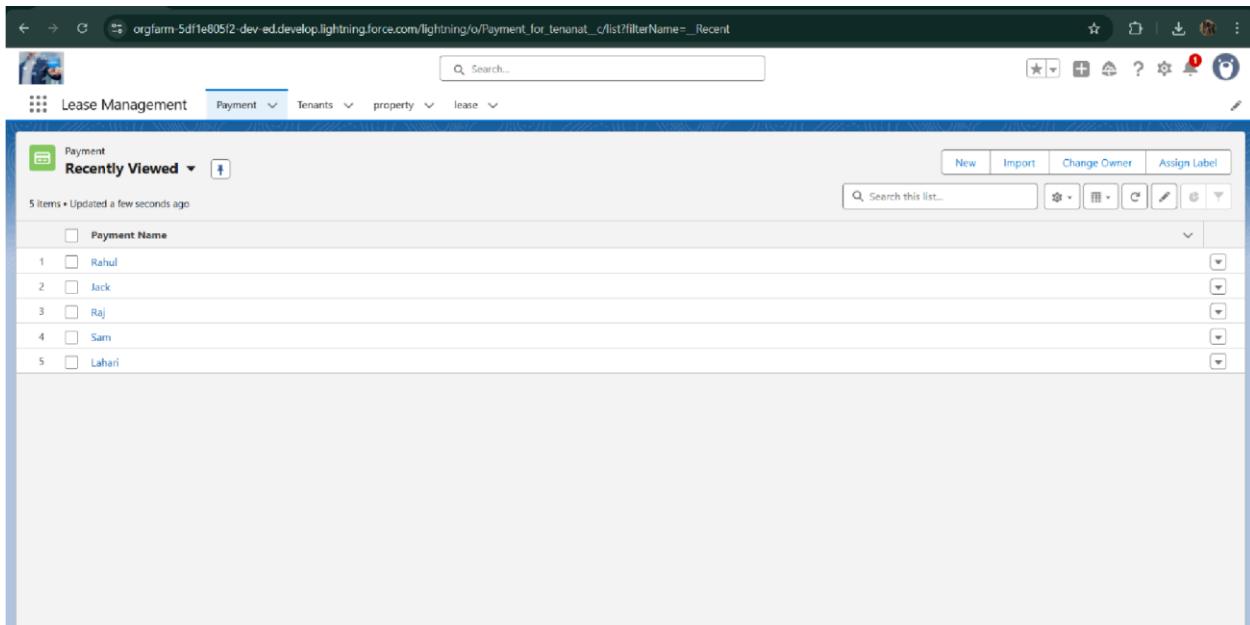
**Available Profiles**

Type to filter list...

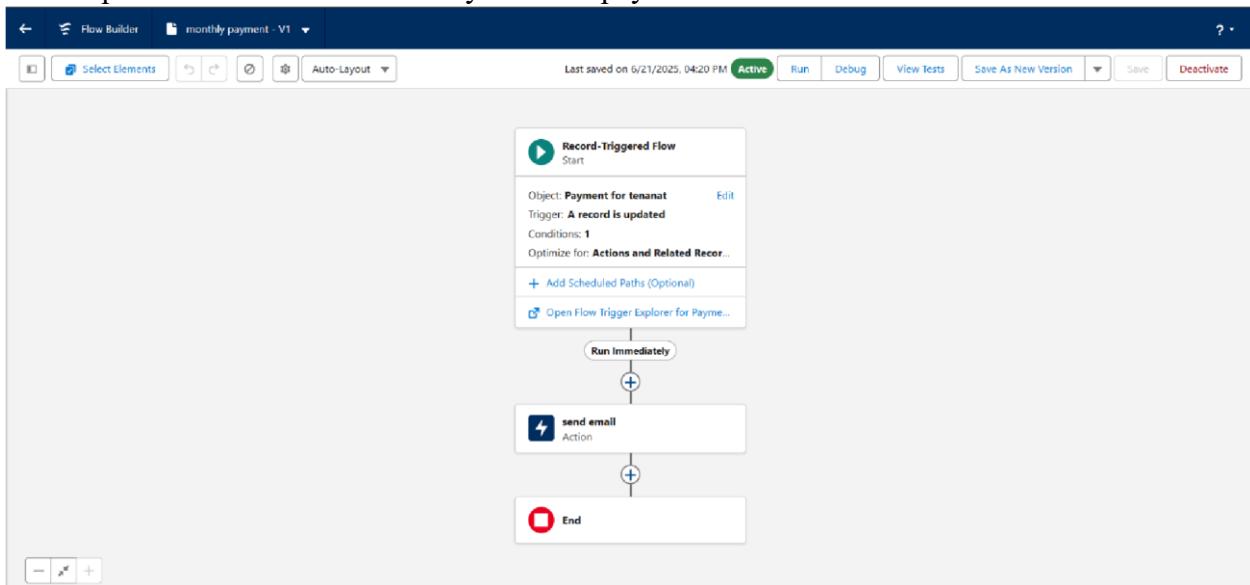
- Analytics Cloud Integration User
- Analytics Cloud Security User
- Anypoint Integration
- Authenticated Website
- Authenticated Website
- B2B Reordering Portal Buyer Profile
- Contract Manager
- Custom: Marketing Profile
- Custom: Sales Profile
- Custom: Support Profile
- Customer Community Login User

**Selected Profiles**

- System Administrator



- Implemented Flows for monthly rent and payment success



- To create a validation rule to a Lease Object

SETUP > OBJECT MANAGER  
lease

**Validation Rule Edit**

Rule Name: lease\_end\_date

Active:

Description:

Error Condition Formula

Example: Discount\_Percent\_C>0.30 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

Insert Field Insert Operator End\_date\_c <= start\_date\_c

Functions

- All Function Categories -
- ABS
- ACOS
- ADDMONTHS
- AND
- ASCII
- ASIN

Insert Selected Function  
ABS(number)  
Returns the absolute value of a number, a number without its sign  
Help on this function

Check Syntax

Save Save & New Cancel

Quick Tips
 

- Operators & Functions

SETUP > OBJECT MANAGER  
lease

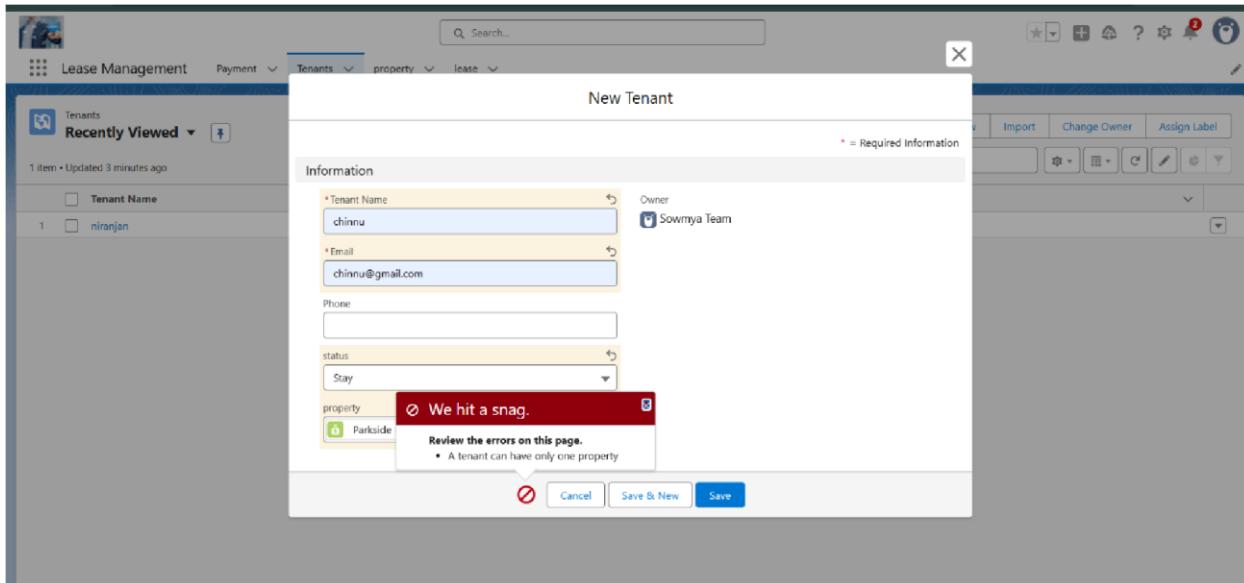
**lease Validation Rule**

Validation Rule Detail

Rule Name	lease_end_date	Active	<input checked="" type="checkbox"/>
Error Condition Formula	End_date_c <= start_date_c	Error Message	Your End date must be greater than start date
Description			
Created By	Sowmya Team	Created Date	6/19/2025, 5:37 AM
Modified By	Sowmya Team	Modified Date	6/26/2025, 7:47 AM

Help for this Page

- Added Apex trigger to restrict multiple tenants per 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 public static void sendMonthlyEmails() {
16
17     List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
18
19     for (Tenant__c tenant : tenants) {
20
21         String recipientEmail = tenant.Email__c;
22
23         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";
24
25         String emailSubject = "Reminder: Monthly Rent Payment Due";
26
27         Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
28
29         email.setToAddresses(new String[]{recipientEmail});
30
31         email.setSubject(emailSubject);
32
33         email.setPlainTextBody(emailContent);
34
35     }
36 }
```

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

The screenshot shows the Salesforce Setup interface with the search bar set to "email template". Under the "Email" category, "Classic Email Templates" is selected. A search result for "Leave approved" is displayed. The "Email Template Detail" section shows the following information:

- Email Template Name:** Leave approved
- Template Unique Name:** Leave\_approved
- Encoding:** Unicode (UTF-8)
- Author:** Sowmya Team [Chennai]
- Description:** (empty)
- Created By:** Sowmya Team, 6/20/2025, 1:06 AM
- Modified By:** Sowmya Team, 6/20/2025, 1:06 AM
- Available For Use:** checked
- Last Used Date:** (empty)
- Times Used:** (empty)

The "Email Template" preview area contains the following content:

```

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.  

your leave is approved. You can leave now.
  
```

The screenshot shows the Salesforce Setup interface with the search bar set to "email template". Under the "Email" category, "Classic Email Templates" is selected. A search result for "tenant leaving" is displayed. The "Email Template Detail" section shows the following information:

- Email Template Name:** tenant leaving
- Template Unique Name:** tenant\_leaving
- Encoding:** Unicode (UTF-8)
- Author:** Sowmya Team [Chennai]
- Description:** (empty)
- Created By:** Sowmya Team, 6/20/2025, 1:06 AM
- Modified By:** Sowmya Team, 6/20/2025, 1:06 AM
- Available For Use:** checked
- Last Used Date:** (empty)
- Times Used:** (empty)

The "Email Template" preview area contains the following content:

```

Subject: request for approve the leave
Plain Text Preview:
Dear {Tenant__c.CreatedBy},  

Please approve my leave
  
```

The screenshot shows the Salesforce Setup interface with the search bar set to "email template". The main content area displays the "Classic Email Templates" page for a "Text Email Template" named "Leave rejected".

**Email Template Detail:**

Email Templates from Salesforce	Unified Public Classic Email Templates
Email Template Name	Leave rejected
Template Unique Name	Leave_rejected
Encoding	Unicode (UTF-8)
Author	Sowmya_Team [Change]
Description	
Created By	Sowmya_Team 6/20/2025, 1:11 AM
Modified By	Sowmya_Team 6/20/2025, 1:11 AM

**Email Template Preview:**

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 search bar set to "email template". The main content area displays the "Classic Email Templates" page for a "Text Email Template" named "Tenant Email".

**Email Template Detail:**

Email Templates from Salesforce	Unified Public Classic Email Templates
Email Template Name	Tenant Email
Template Unique Name	Tenant_Email
Encoding	Unicode (UTF-8)
Author	Sowmya_Team [Change]
Description	
Created By	Sowmya_Team 6/20/2025, 1:12 AM
Modified By	Sowmya_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.

This communication is a friendly reminder concerning your monthly rent payment. When it's due, please make sure to pay it.

**Email Template Detail**

- Email Template Name: tenant payment
- Unique Name: tenant\_payment
- Encoding: Unicode (UTF-8)
- Author: Sowmya Team (Change)
- Created By: Sowmya Team, 6/20/2025, 1:13 AM
- Modified By: Sowmya 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:

**Process Definition Detail**

- Process Name: TenantApproval
- Unique Name: TenantApproval
- Description: Tenant: status equals Stay
- Entry Criteria: DEBAND1: status equals Stay
- Record Editability: Administrator ONLY
- Initial Submitter: Tenant Owner
- Created By: Sowmya Team, 6/25/2025, 3:41 AM
- Modified By: 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	1	Step 1			User_Sowmya Team	Final Rejection

For Check for Vacant:

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

- Approval Processes:** Tenant: check for vacant
- Process Definition Detail:**
  - Process Name: check for vacant
  - Unique Name: Check\_for\_vacant
  - Description:
  - Entry Criteria: Tenant\_\_c: status EQUALS Leaving
  - Record Visibility: Administrator ONLY
  - Next Automated Approver Determined By: Allow Submitters to Recall Approval Requests
  - Approval Assignment Email Template: Leave\_approved
  - Initial Submitters: Tenant Owner
  - Created By: Sowmya\_Team
  - Modified By: Sowmya\_Team
  - Active: Yes
- Initial Submission Actions:**
  - Action: Type: Record Lock
  - Description: Lock the record from being edited
  - Action: Type: Email Alert
  - Description: 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

```

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

```

The screenshot shows the Salesforce Developer Console with the following details:

- Trigger:** test.apex
- Code:**

```

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

```
- Modal Dialog:** Open dialog for selecting an entity type, showing options like Classes, Triggers, Pages, etc.
- Toolbar:** Logs, Tests, Checkpoints, Query Editor, View State, Progress, Problems

Developer Console - Google Chrome  
 orgfarm-5d1fe805f2-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

```
File • Edit • Debug • Test • Workspace • Help • < >
test.apex [ testHandler.apxc ] MonthlyEmailScheduler.apxc [ ]
Code Coverage: None • API Version: 64 • Go To
1 trigger test on Tenant__c (before insert)
2
3 +
4 if(trigger.isInsert && trigger.isBefore){
5
6   testHandler.preventInsert(trigger.new);
7
8 }
9
10
11 }
```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

## Create an Apex Handler class

Developer Console - Google Chrome  
 orgfarm-5d1fe805f2-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

```
File • Edit • Debug • Test • Workspace • Help • < >
test.apex [ testHandler.apxc ] MonthlyEmailScheduler.apxc [ ]
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(exi
10      }
11
12
13
14+     for (Tenant__c newTenant : newList) {
15
16
17       if (newTenant.Property__c != null) {
18
19         newTenantaddError('A t
20
21
22     }
23 }
```

Entity Type Entity Name Namespace Related

Entity Type	Entity	Related
Classes	testHandler	← test ApexTrigger References...
Triggers	MonthlyEmailScheduler	← property CustomField References...
Pages		← Tenant__c Object References...
Objects		← Tenant__c Object References...
Static Resources		
Packages		

Open Filter Hide Managed Packages Refresh

Logs Tests Checkpoints Query Editor View State Progress Problems

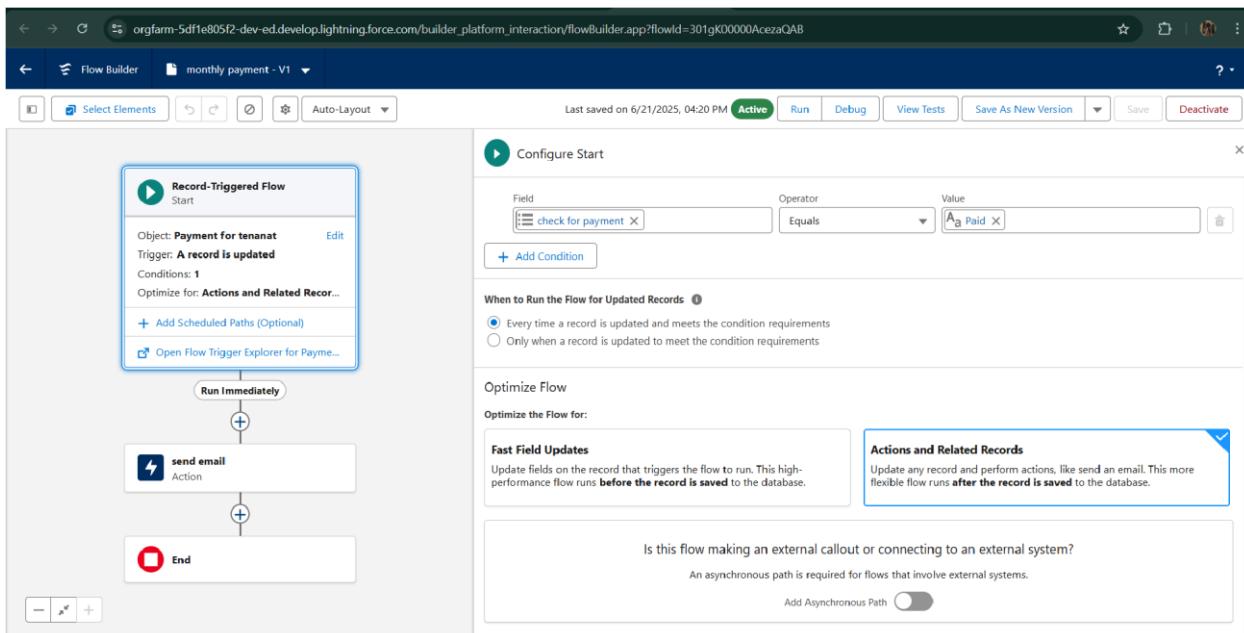
Name Line Problem

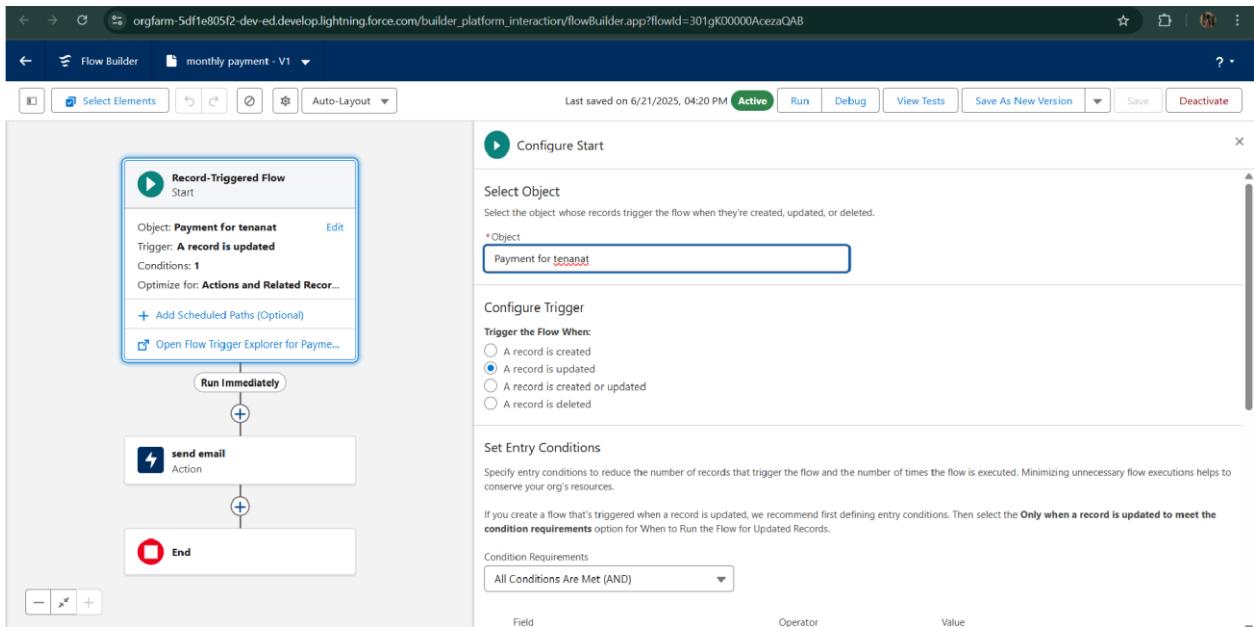
```

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         }
24
25     }
26
27 }

```

## ● FLOWS





- Schedule class:  
Create an 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 sendMonthlyEmail
17
18     List<Tenant__c> tenants = [SEL
19
20         for (Tenant__c tenant : tenants) Open Filter Hide Managed Packages Refresh
21
22             String recipientEmail = tenant.Email__c;
23

```

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

test.apex testHandler.apex [ MonthlyEmailScheduler.apex ] Go To

Code Coverage: None API Version: 64

Open

Entity Type	Entities	Related
Classes	testHandler	Name Extent Direction
Triggers	MonthlyEmailScheduler	Email CustomField... References
Pages		← Tenant__c Object References
Page Components		← Tenant__c Subject References
Objects		
Static Resources		
Packages		

Logs Tests Checkpoints Query Editor View Status Progress Problems

Developer Console - Google Chrome

orgfarm-5d1fe05f2-dev-ed.develop.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

```

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 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             email.setSubject(emailSubject);
32             email.setPlainTextBody(emailContent);
33
34             messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
35
36         }
37
38     }
39
40 }
41

```

## Schedule Apex class

Setup | Home | Components | Query Editor | View Home | Logout | Help

Search Setup

Apex Classes

Apex Class Detail

Name: MonthlyEmailScheduler

Namespace Prefix:

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

Status: Active

Code Coverage: 0% (0/15)

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

Class Body

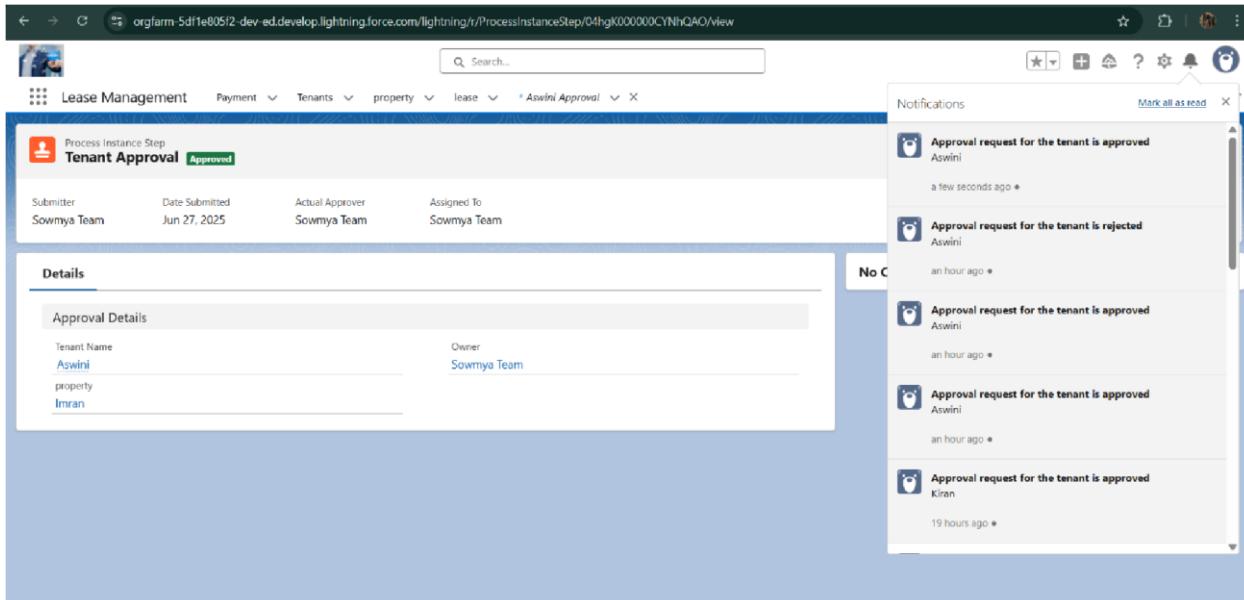
```

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 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             email.setSubject(emailSubject);
32             email.setPlainTextBody(emailContent);
33
34             messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
35
36         }
37
38     }
39
40 }
41

```

The screenshot shows a Salesforce Lightning interface for a 'Lease Management' application. The main page displays a 'Tenant' record for 'Aswini'. The 'Details' tab is selected, showing fields for Tenant Name (Aswini), Email (aswiniamaraadi15@gmail.com), Phone ((905) 223-5567), Status (Leaving), and Property (Imran). The record was created by 'Sowmya Team' on 6/26/2025 at 6:05 AM and last modified by 'Sowmya Team' on 6/26/2025 at 11:06 PM. A sidebar titled 'Activity' is open, showing a context menu with options like 'New Case', 'New Lead', 'Delete', 'Clone', 'Change Owner', 'Printable View', 'Submit for Approval', and 'Edit Labels'. The 'Upcoming & Overdue' section indicates 'No activities to show.'

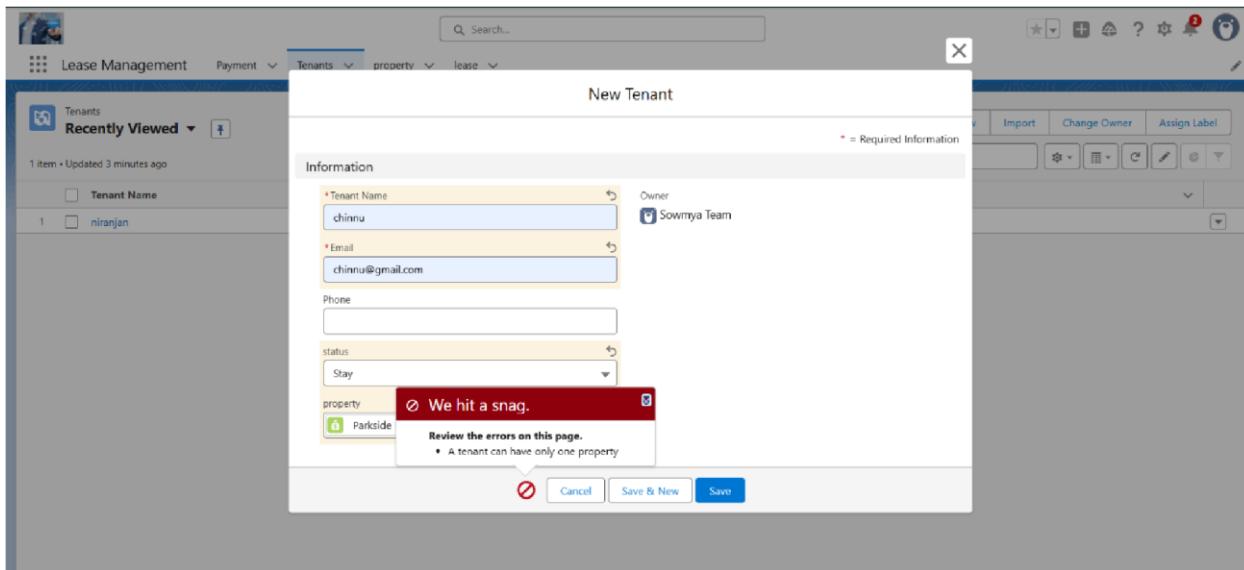
The screenshot shows the same Salesforce Lightning interface after the record has been submitted for approval. A green success message 'Tenant was submitted for approval.' is displayed above the activity sidebar. The rest of the interface remains identical to the first screenshot, showing the tenant details and the 'Activity' sidebar.



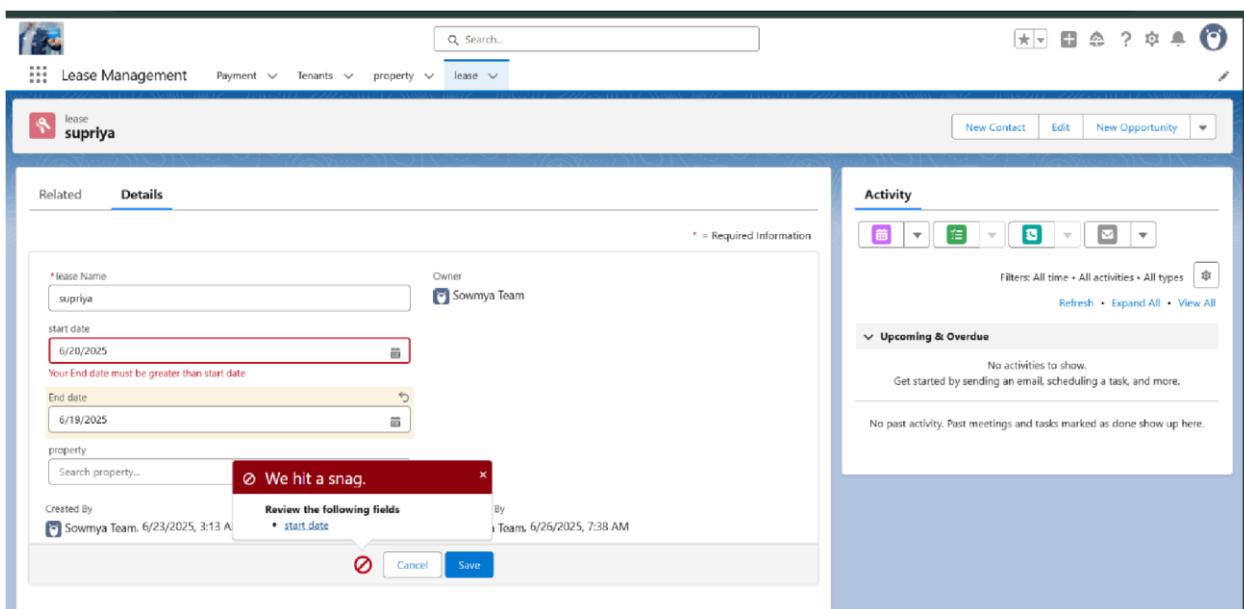
# FUNCTIONAL AND PERFORMANCE TESTING

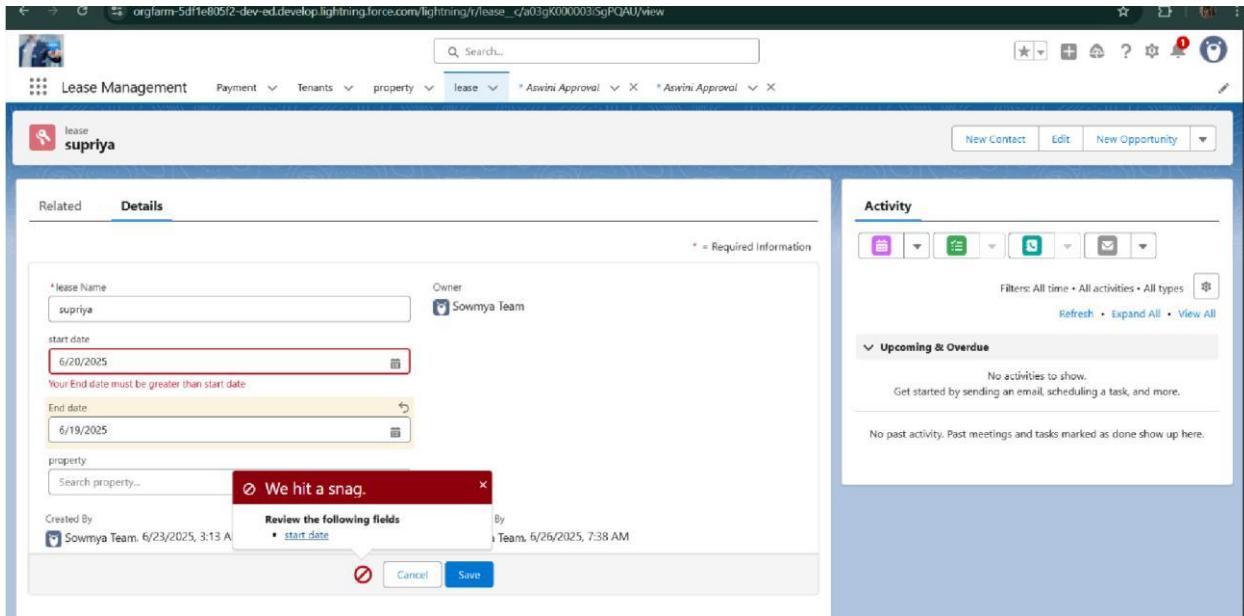
## Performance Testing

- Trigger validation by entering duplicate tenant-property records

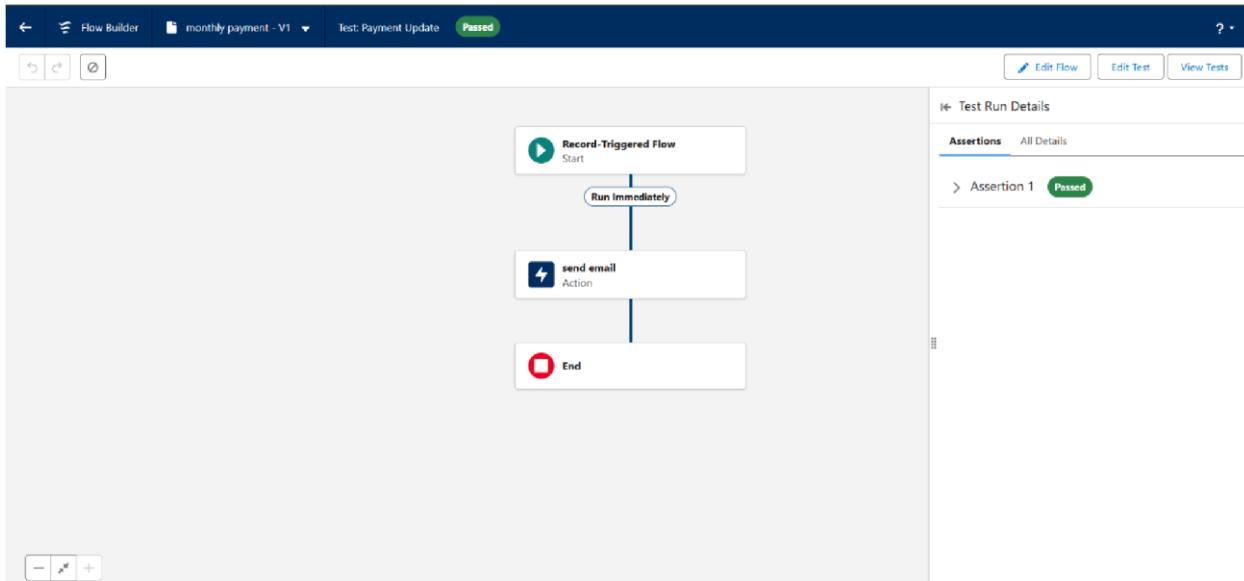


- Validation Rule checking





- Test flows on payment update



- Approval process validated through email alerts and status updates

The screenshot shows a Microsoft Dynamics 365 Lease Management tenant record for "niranjan". The main pane displays the tenant's details: Name (niranjan), Email (niranjan1506@gmail.com), Owner (Sowmya Team), Phone (empty), Status (Stay), and Property (Parkside Lofts). The record was created by Sowmya Team on June 23, 2025, at 2:33 AM, and last modified by Sowmya Team on June 23, 2025, at 3:58 AM. A sidebar titled "Notifications" lists several recent events:

- 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) - Jun 20, 2025, 1:28 PM

The screenshot shows a Microsoft Dynamics 365 Lease Management tenant record for "niranjan". The main pane displays the tenant's details and includes sections for "Approval History" and "Payment".

**Approval History (6+)**

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

**Payment (2)**

Payment Name
Jack
Rahul

A sidebar on the right indicates no past activity, past meetings, or tasks marked as done.

# RESULTS

## Output Screenshots

- Tabs for Property, Tenant, Lease, Payment

The screenshot shows the Salesforce Setup interface under the 'Tabs' section. It displays a table of 'Custom Object Tabs' with columns for Action, Label, Tab Style, and Description. The tabs listed are:

Action	Label	Tab Style	Description
Edit   Del	Lease	Keys	
Edit   Del	Payment	Credit card	
Edit   Del	property	Sack	
Edit   Del	Tenant	Map	

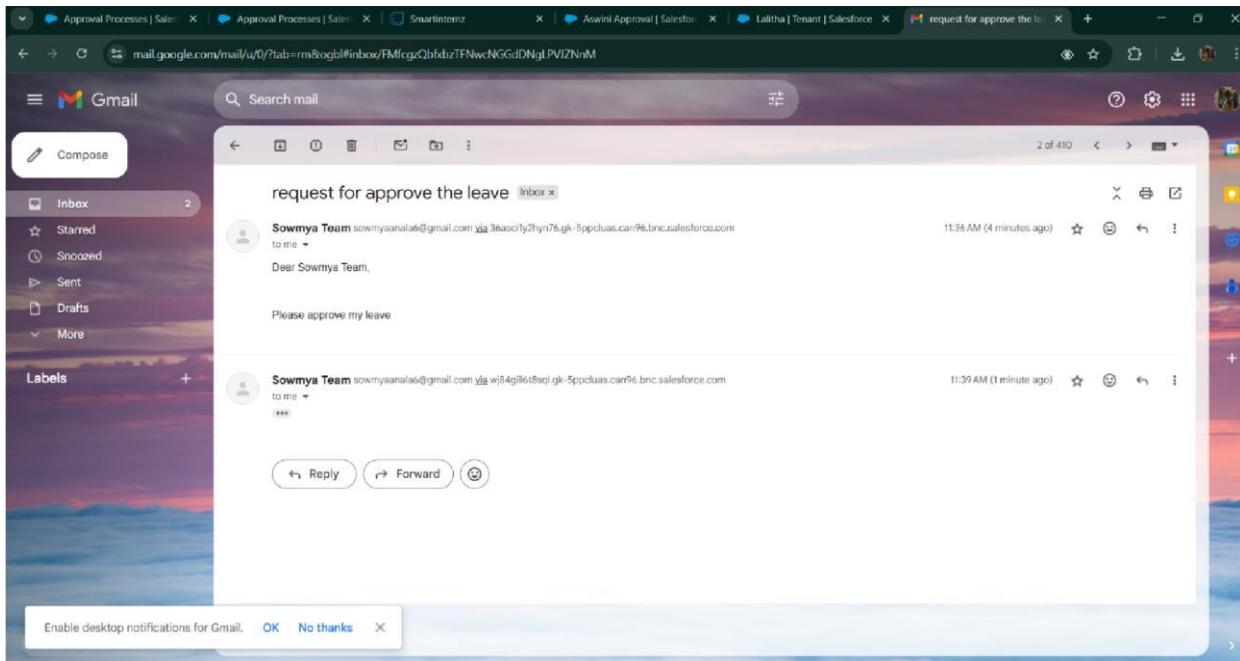
Below this are sections for 'Web Tabs' and 'Visualforce Tabs', both of which currently have no tabs defined.

- Email alerts

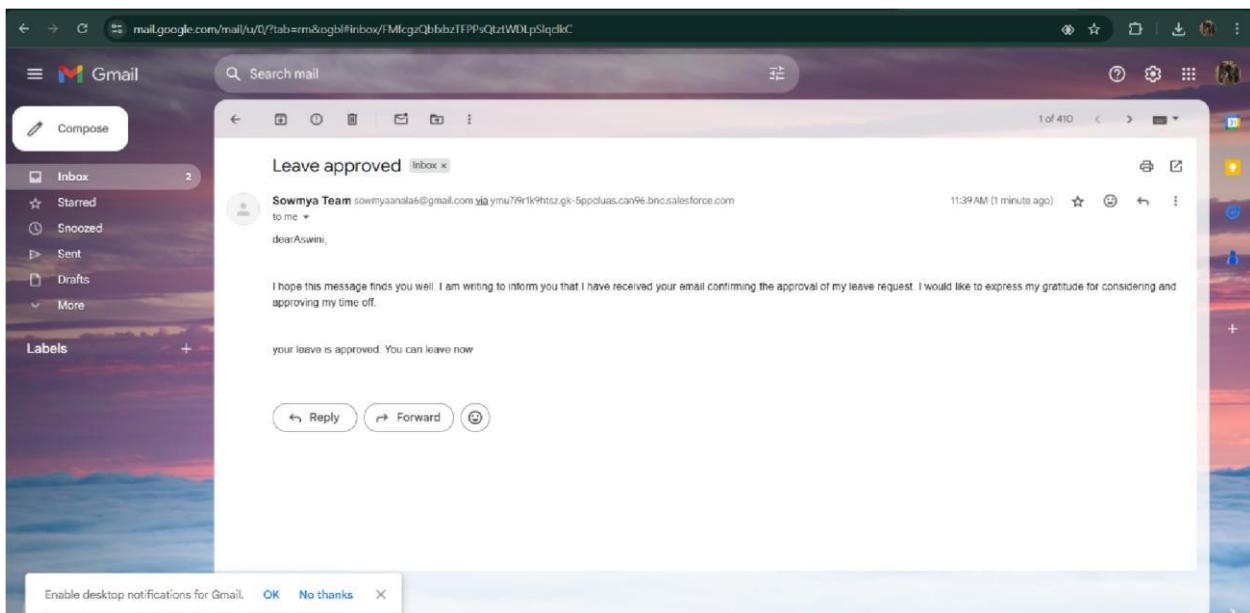
The screenshot shows the 'Lease Management' application interface. A navigation bar at the top includes 'Lease Management', 'Payment', 'Tenants', 'property', and 'lease'. The main area is titled 'Approval History' and shows a table of approval steps for a leave request. The table has columns for Step Name, Date, Status, Assigned To, Actual Approver, and Comments.

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

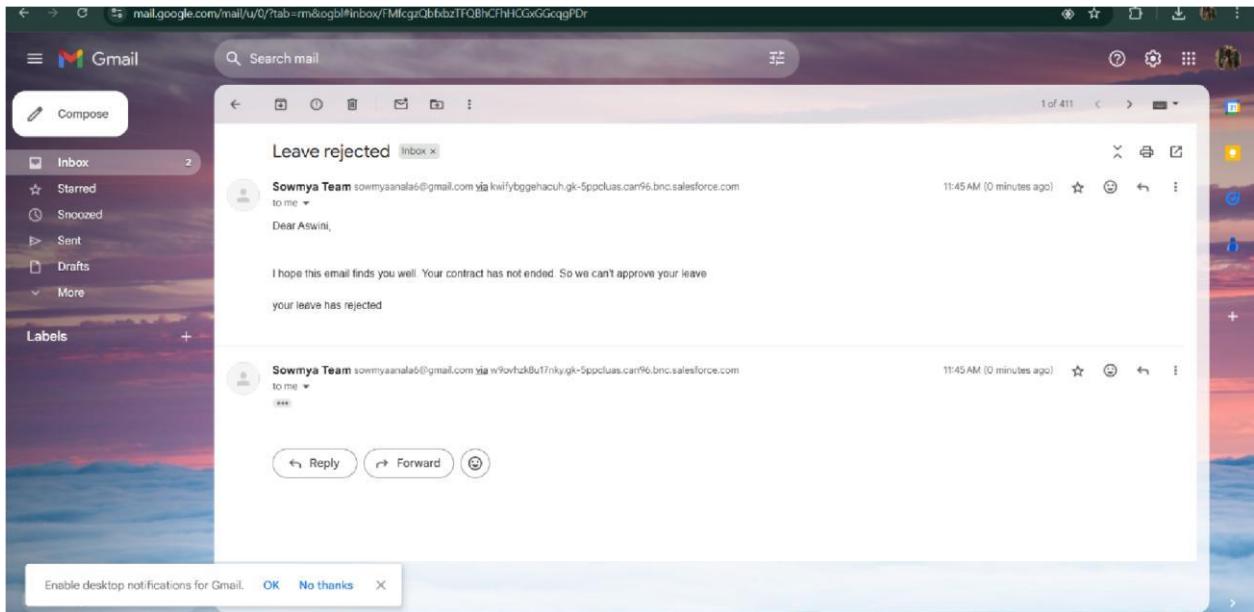
- Request for approve the leave



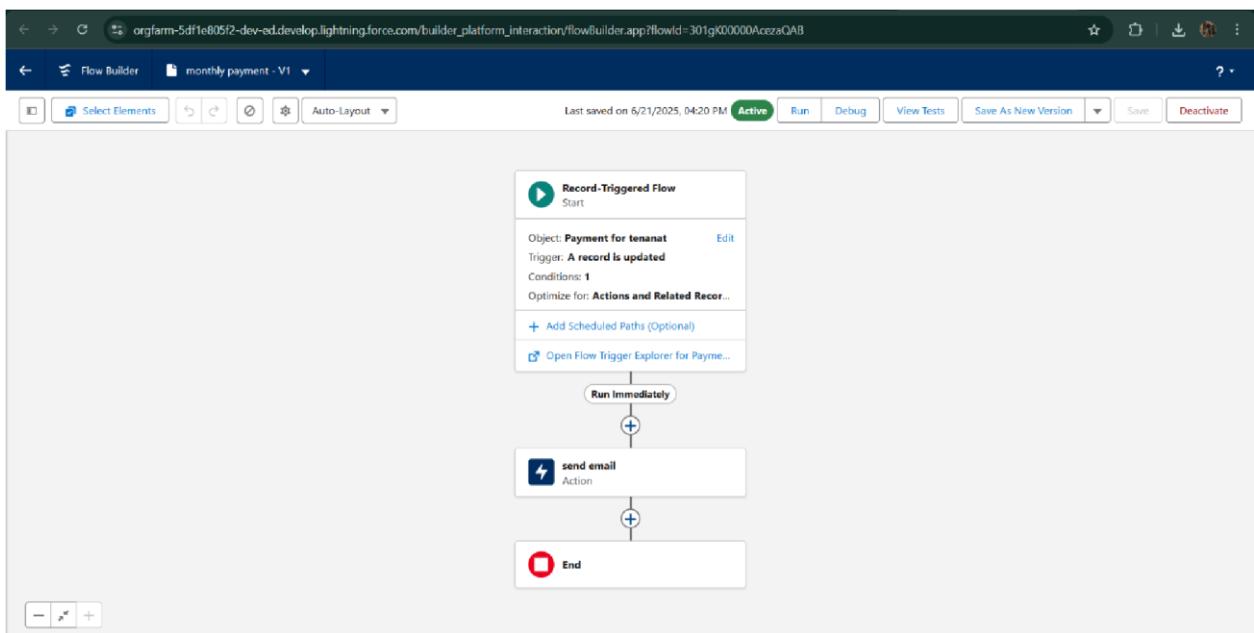
- Leave approved



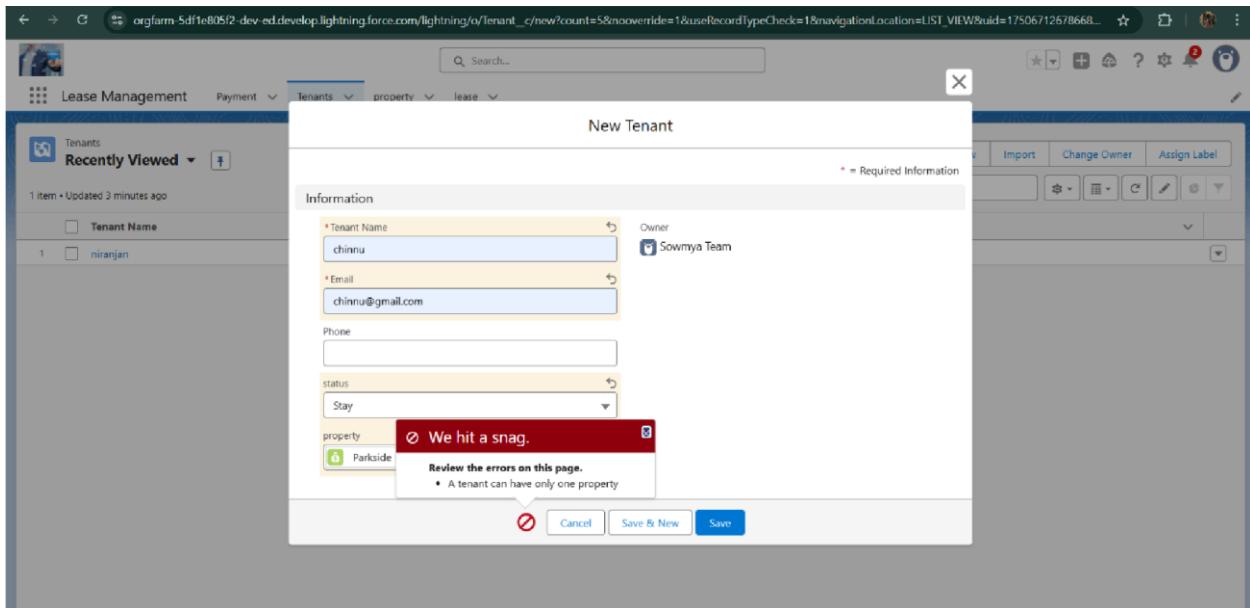
- Leave rejected



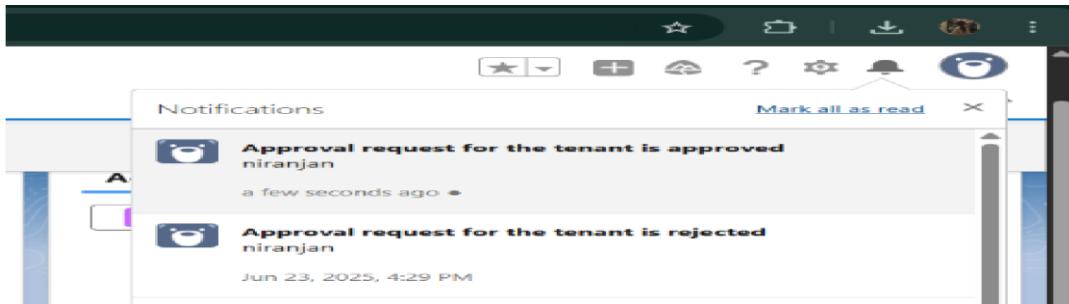
- Flow runs



- Trigger error messages



- Approval process notifications



## ADVANTAGES & DISADVANTAGES

---

# 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.

---

## 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});  
    }  
}  
}
```