

LEASE MANAGEMENT

College Name: Shree Venkateshwara Arts and Science(Co-Education) College
College Code: brubd

TEAM ID: NM2025TMID20958

TEAM MEMBERS:

Team LeaderName:CHANDRU OP

Email:chandruop2023aids@gmail.com

Team Member1: DINESH J

Email:dineshj2023aids2@gmail.com

Team Member2: JEEVANANDHAKUMAR K

Email:jeevanandhakumark2023aids@gmail.com

Team Member3: DINESH R

Email:dineshr2023aids@gmail.com

Team Member4: JEEVABHARATHI S

Email:jeevabharathis2023aids@gmail.com

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.

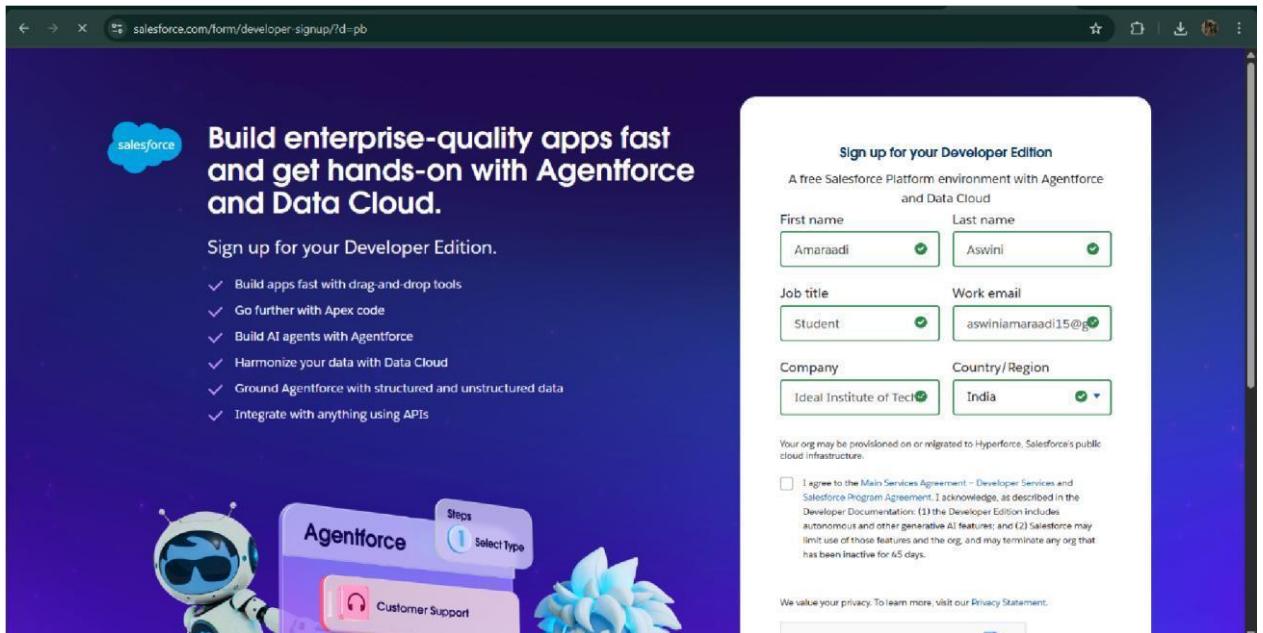
1.2 Purpose

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

Details	
API Name	property__c
Custom	<input checked="" type="checkbox"/>
Singular Label	property
Plural Label	property
Description	
Enable Reports	
<input checked="" type="checkbox"/> Track Activities	
<input checked="" type="checkbox"/> Track Field History	
<input checked="" type="checkbox"/> Deployment Status	
Deployed	
Help Settings	
Standard salesforce.com Help Window	

SETUP > OBJECT MANAGER

Tenant

Details

Description

API Name: Tenant__c

Custom

Singular Label: Tenant

Plural Label: Tenants

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

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

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Setup', 'Home', 'Object Manager', and a search bar. The main title is 'SETUP > OBJECT MANAGER Payment for tenant'. On the left, a sidebar lists various configuration options under 'Details' and 'Fields & Relationships'. The main content area displays the 'Details' tab for the 'Payment for tenant' object. It shows the API Name as 'Payment_for_tenant_c', a custom singular label 'Payment for tenant', and a plural label 'Payment'. To the right, there are sections for 'Enable Reports' (checkboxes for Track Activities, Track Field History, Deployment Status, Deployed, Help Settings, and Standard salesforce.com Help Window), and buttons for 'Edit' and 'Delete'.

- Configured fields and relationships

The screenshot shows the Salesforce Object Manager interface for the 'property' object. The top navigation bar includes 'Setup', 'Home', 'Object Manager', and a search bar. The main title is 'SETUP > OBJECT MANAGER property'. On the left, a sidebar lists various configuration options under 'Details' and 'Fields & Relationships'. The main content area displays the 'Fields & Relationships' tab, which lists 9 items sorted by Field Label. A table provides detailed information for each field:

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)	<input checked="" type="checkbox"/>	
property	property__c	Lookup(property)	<input checked="" type="checkbox"/>	
property Name	Name	Text(80)	<input checked="" type="checkbox"/>	
sfqt	sfqt__c	Text(18)		
Type	Type__c	Picklist		

Setup > Object Manager

Payment for tenant

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

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedBy	Lookup(User)		
End date	End_date_c	Date		
Last Modified By	LastModifiedById	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	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

Lightning App Builder

App Settings

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:  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

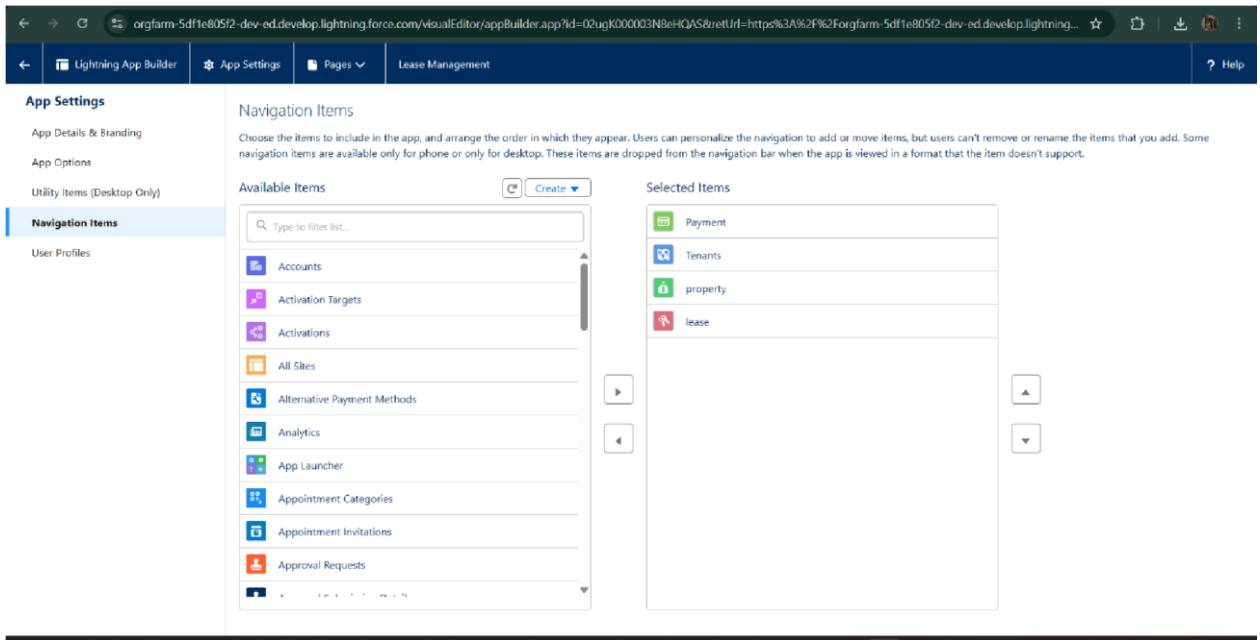
Type to filter list... [Create](#)

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

Selected Items

- Payment
- Tenants
- property
- lease

Up ▲ Down ▼



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

App Settings

[App Details & Branding](#)

[App Options](#)

[Utility Items \(Desktop Only\)](#)

User Profiles

User Profiles

Choose the user profiles that can access this app.

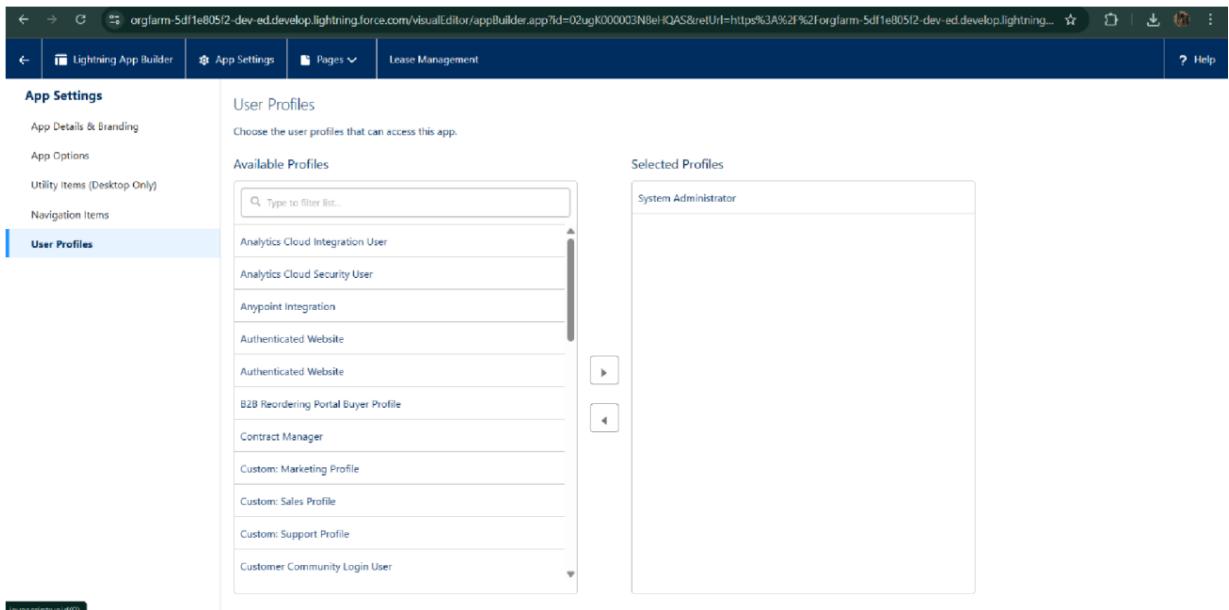
Available Profiles

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

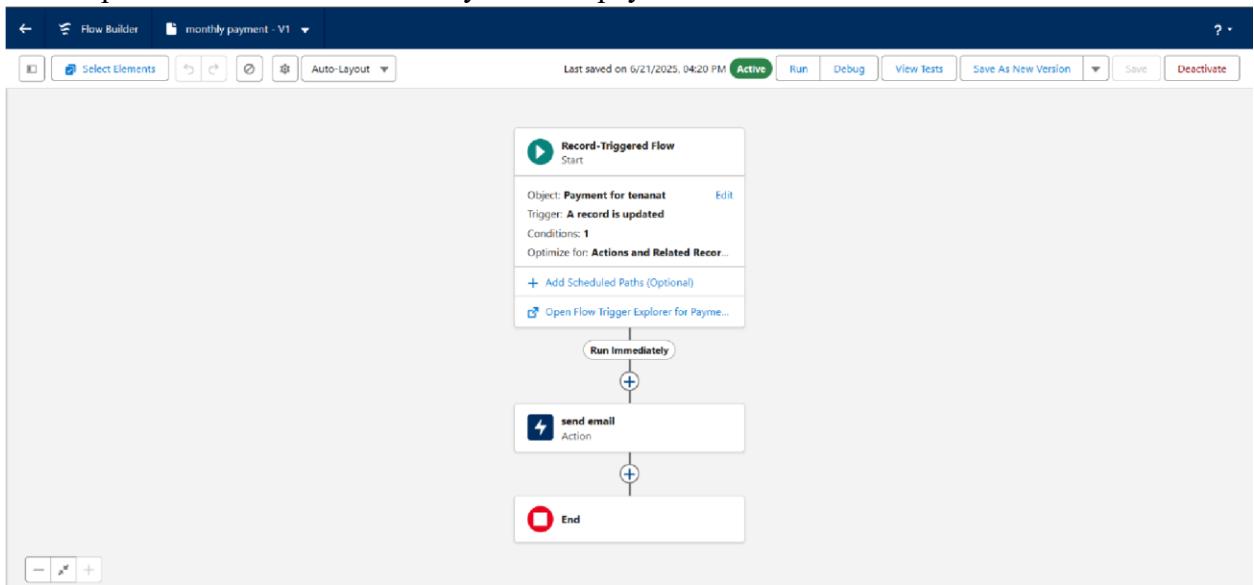
Selected Profiles

- System Administrator



The screenshot shows the Salesforce Lease Management interface. The top navigation bar includes tabs for 'Lease Management', 'Payment', 'Tenants', 'property', and 'lease'. A search bar is at the top right. Below the navigation is a section titled 'Recently Viewed' with a 'Payment' icon. The main content area displays a list of 5 items under 'Payment Name': 1. Rahul, 2. Jack, 3. Raj, 4. Sam, 5. Lahari. To the right of the list are several small icons for actions like New, Import, Change Owner, and Assign Label.

- Implemented Flows for monthly rent and payment success



- To create a validation rule to a Lease Object

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

- Page Header:** orgfarm-5d1fe805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK00000zTujValidationRules/page?address=%2F03dgK00000C8qPQAG%2Fe%3freURL%3D%2...
- Page Title:** SETUP > OBJECT MANAGER
- Object:** lease
- Section:** Validation Rule Edit
- Fields:**
 - Rule Name: lease_end_date
 - Active: checked
 - Description: (empty)
- Error Condition Formula:**

```
Example: Discount_Percent__c>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
```

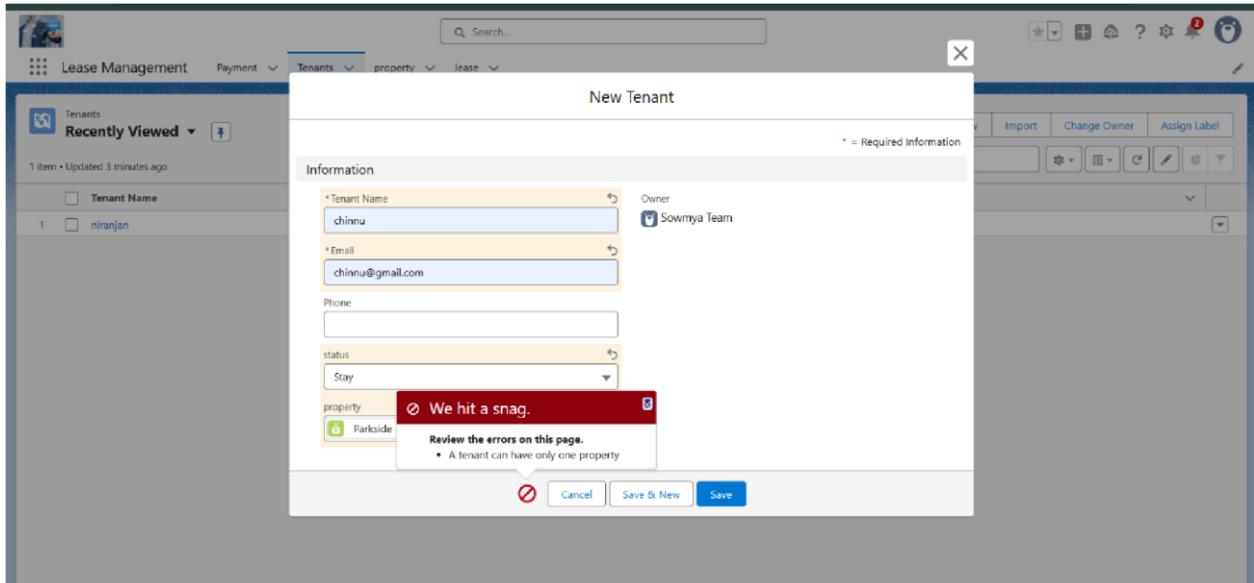
Formula: End_date_c <= start_date_c
- Functions:**
 - All Function Categories
 - ABS
 - ACOS
 - ADDMONTHS
 - AND
 - ASCII
 - ASIN
- Buttons:**
 - Save
 - Save & New
 - Cancel
 - Check Syntax
- Help:** Quick Tips (Operators & Functions)

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

- Page Header:** orgfarm-5d1fe805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK00000zTujValidationRules/03dgK00000C8qPQAG/view
- Page Title:** SETUP > OBJECT MANAGER
- Object:** lease
- Section:** lease Validation Rule
- Fields:**

Field	Value
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	(empty)
Created By	Sowmya Team 6/19/2025, 5:37 AM
Modified By	Sowmya Team 6/26/2025, 7:47 AM
- Buttons:**
 - Edit
 - Close
- Help:** 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         email.setSubject(emailSubject);
31         email.setPlainTextBody(emailContent);
32
33     }
34 }
35

```

- 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 specific template named 'Leave approved' is displayed in the center. The 'Email Template Detail' section shows the following information:

- Email Templates from Salesforce:** Leave approved
- Unified Public Classic Email Templates:** Leave approved
- Template Unique Name:** Leave_approved
- Encoding:** Unicode (UTF-8)
- Author:** Sowmya Team [Change]
- Description:** (empty)
- Created By:** Sowmya Team, 6/20/2025, 1:08 AM
- Modified By:** Sowmya Team, 6/20/2025, 1:08 AM
- Available For Use:** checked
- Last Used Date:** (empty)
- Times Used:** (empty)

The 'Email Template' preview area shows 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 specific template named 'tenant leaving' is displayed in the center. The 'Email Template Detail' section shows the following information:

- Email Templates from Salesforce:** tenant leaving
- Unified Public Classic Email Templates:** tenant leaving
- Template Unique Name:** tenant_leaving
- Encoding:** Unicode (UTF-8)
- Author:** Sowmya Team [Change]
- 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 shows 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". Under the "Email" category, "Classic Email Templates" is selected. A search result for "Leave rejected" is displayed. The "Email Template Detail" section shows the following information:

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

Buttons for "Edit", "Delete", and "Clone" are available. The "Available For Use" checkbox is checked. The "Email Template" preview section shows the subject "Leave rejected" and the plain text 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

A "Send Test and Verify Merge Fields" button is present.

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 Email" is displayed. The "Email Template Detail" section shows the following information:

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

Buttons for "Edit", "Delete", and "Clone" are available. The "Available For Use" checkbox is checked. The "Email Template" preview section shows the subject "Urgent: Monthly Rent Payment Reminder" and the plain text 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.

A "Send Test and Verify Merge Fields" button is present.

Classic Email Templates

tenant payment

Email Template Detail

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

Email Template

Plain Text Preview

Subject: Confirmation of Successful Monthly Payment

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
 Unique Name: TenantApproval
 Description:
 Entry Criteria: tenant__c.status equals Stay
 Record Editability: Administrator ONLY
 Next Automated Approver Determined By
 Allow Submitters to Recall Approval Requests

Initial Submission Actions

Action Type: Record Lock
 Description: 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** page for "Tenant: check for vacant".
- Process Definition Detail** section:
 - Process Name: check for vacant
 - Unique Name: check_for_vacant
 - Description: Tenant status EQUALS Leaving
 - Entry Criteria: Tenant Owner
 - Record Editability: Administrator ONLY
 - Next Automated Approver Determined By: Active (checked)
 - Approval Assignment Email Template: Leave approved
 - Initial Submitters: Tenant Owner
 - Created By: Sowmya Team
 - Modified By: Sowmya Team
- Initial Submission Actions** section:

Action	Type	Description
Record Lock		Lock the record from being edited
Email Alert		please approve my leave
- Approval Steps** section:

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

● Apex Trigger

Create an Apex Trigger

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

- trigger test on Tenant__c (before insert)**
- Code Coverage**: 0% (API Version: 64)
- Entity Type** search results (Modal):

Entity Type	Entities	Related
Triggers	test	
- Logs**, **Tests**, **Checkpoints**, **Query Editor**, **View State**, **Progress**, **Problems** tabs at the bottom.

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

File Edit Debug Test Workspace Help < >

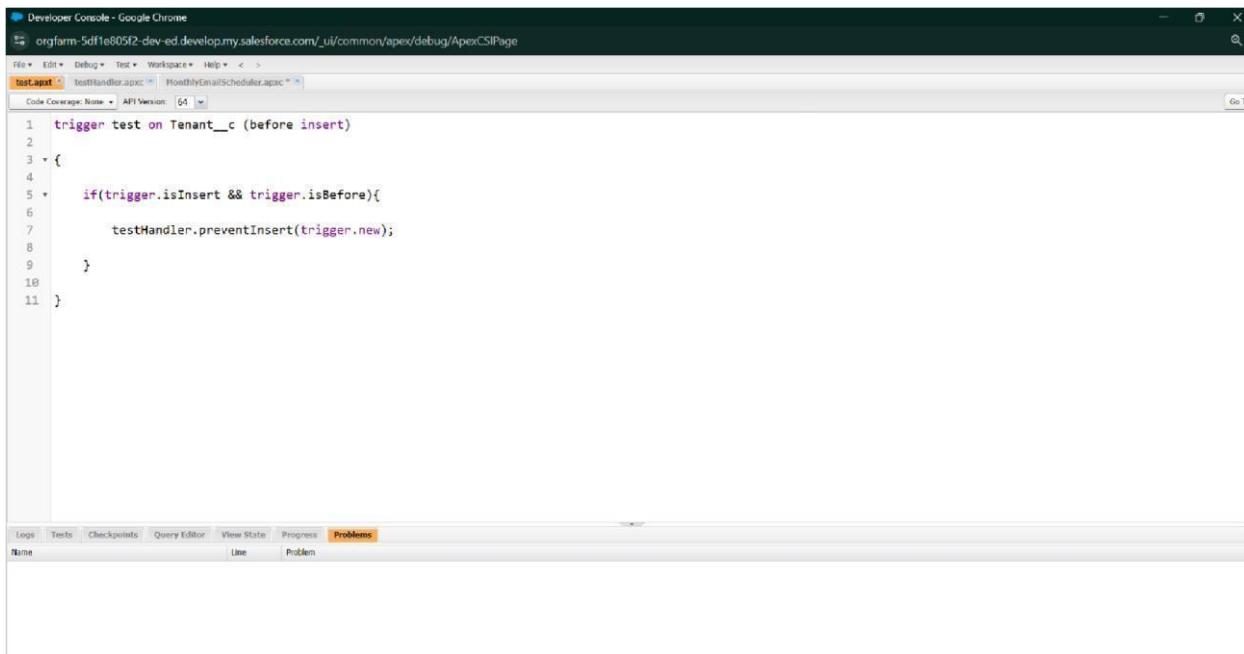
test.apex [] MonthlyEmailScheduler.apac []

Code Coverage Name: 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

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 [] MonthlyEmailScheduler.apac []

Code Coverage Name: API Version: 64 Go To

```
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.Id);
        }
        for (Tenant__c newTenant : newList) {
            if (newTenant.Property__c != null) {
                newTenantaddError('A');
            }
        }
    }
}
```

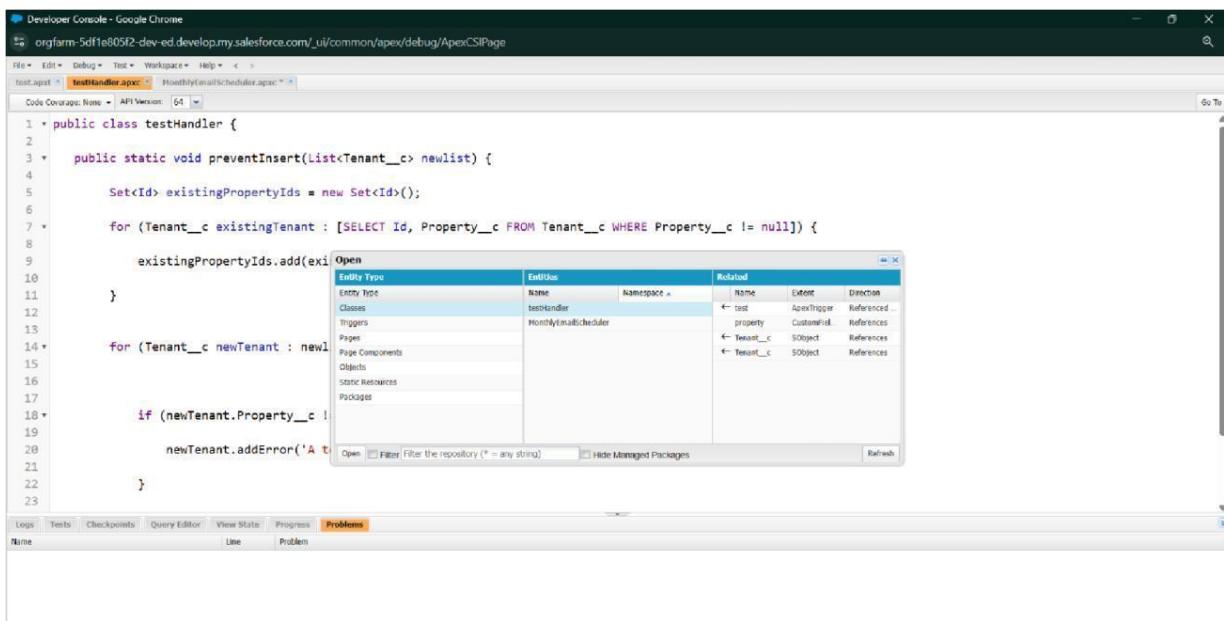
Open Entity Type Entities Related

Entity Type	Entities	Related
Apex Trigger	testHandler	test (ApexTrigger)
Classes	MonthlyEmailScheduler	property (CustomField)
Triggers		Tenant__c (Object)
Pages		Tenant__c (Object)
Page Components		
Objects		
Static Resources		
Packages		

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

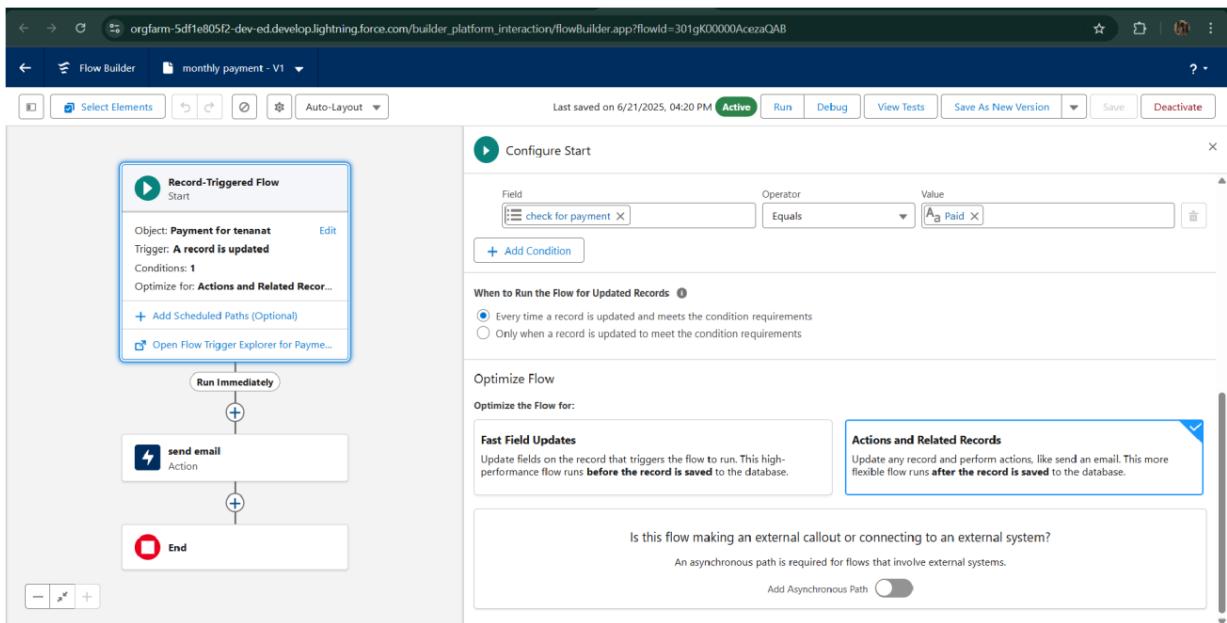
Logs Tests Checkpoints Query Editor View State Progress Problems

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 }
```

- 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 = [SELECT
19
20         for (Tenant__c tenant : tenants
21
22             String recipientEmail = tenant.Email__c;
23

```

```
Developer Console - Google Chrome
http://orgfarm-5df1e05f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
50% Reset

global class MonthEmailScheduler 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.setToRecipients(new String[]{recipientEmail});
        email.setSubject(emailSubject);
        email.setPlainTextBody(emailContent);
        Messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
    }
}

```

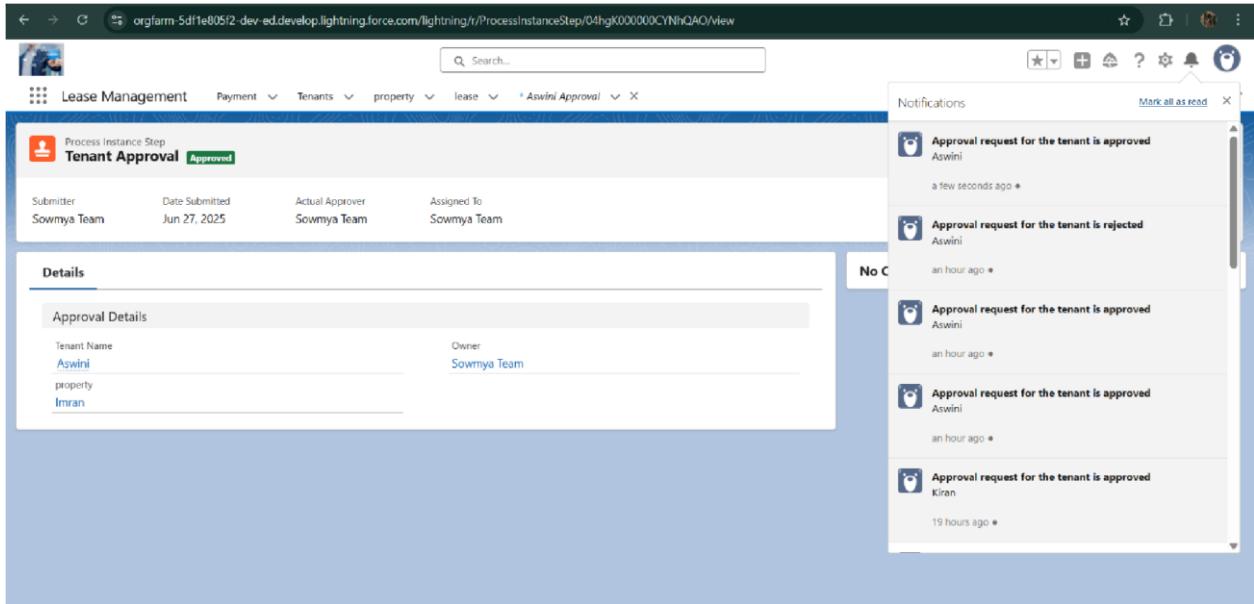
Schedule Apex class

The screenshot shows the Salesforce Setup interface with the 'Apex Classes' tab selected. The page title is 'Apex Classes' under the 'SETUP' section. Below it, the class name 'MonthlyEmailScheduler' is displayed. The 'Apex Class Detail' section includes fields for Name (MonthlyEmailScheduler), Namespace Prefix (Sowmya_Team), Created By (Sowmya Team), Status (Active), Code Coverage (0% (015)), and Last Modified By (Sowmya Team). Below this, tabs for 'Class Body', 'Class Summary', 'Version Settings', and 'Trace Flags' are visible. The 'Class Body' tab is active, showing the following Apex code:

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

The screenshot shows a Salesforce Lightning page for a tenant named 'Aswini'. The page has a header with tabs for 'Lease Management', 'Payment', 'Tenants', 'property', and 'lease'. A search bar is at the top right. On the left, there's a sidebar with 'Related' and 'Details' tabs. The 'Details' tab is active, showing fields for 'Tenant Name' (Aswini), 'Email' (aswiniamaraadi15@gmail.com), 'Phone' ((905) 223-5567), 'status' (Leaving), and 'property' (Imran). It also shows 'Created By' (Sowmya Team) and 'Last Modified By' (Sowmya Team). On the right, there's an 'Activity' sidebar with options like 'New Case', 'New Lead', 'Delete', 'Clone', 'Change Owner', 'Printable View', 'Submit for Approval', and 'Edit Labels'. A message says 'No activities to show.' and 'Get started by sending an email, scheduling a task, and more.'

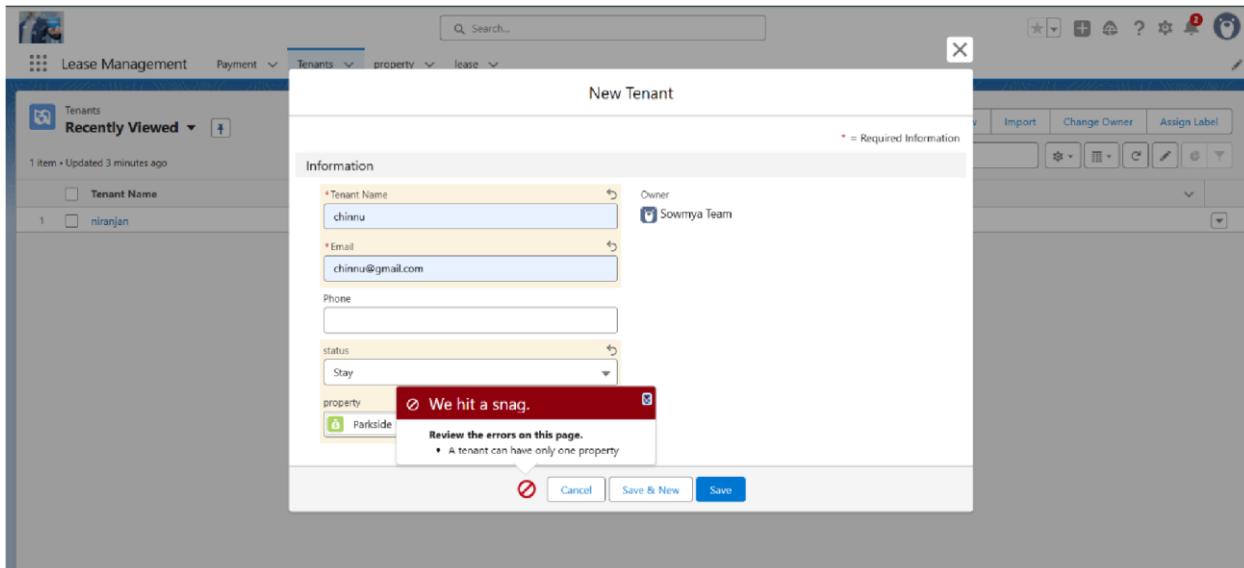
This screenshot shows the same Salesforce page after the 'Submit for Approval' button was clicked. A green success message at the top right says 'Tenant was submitted for approval.' The rest of the page remains identical to the first screenshot, showing the tenant details and the empty activity sidebar.



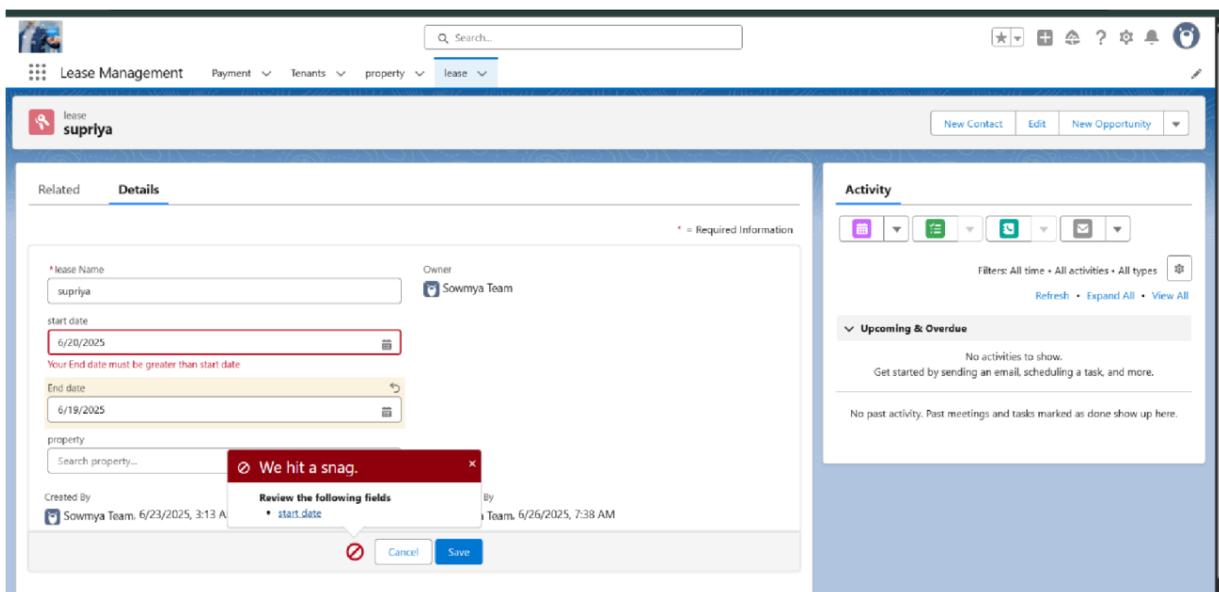
FUNCTIONAL AND PERFORMANCE TESTING

Performance Testing

- Trigger validation by entering duplicate tenant-property records

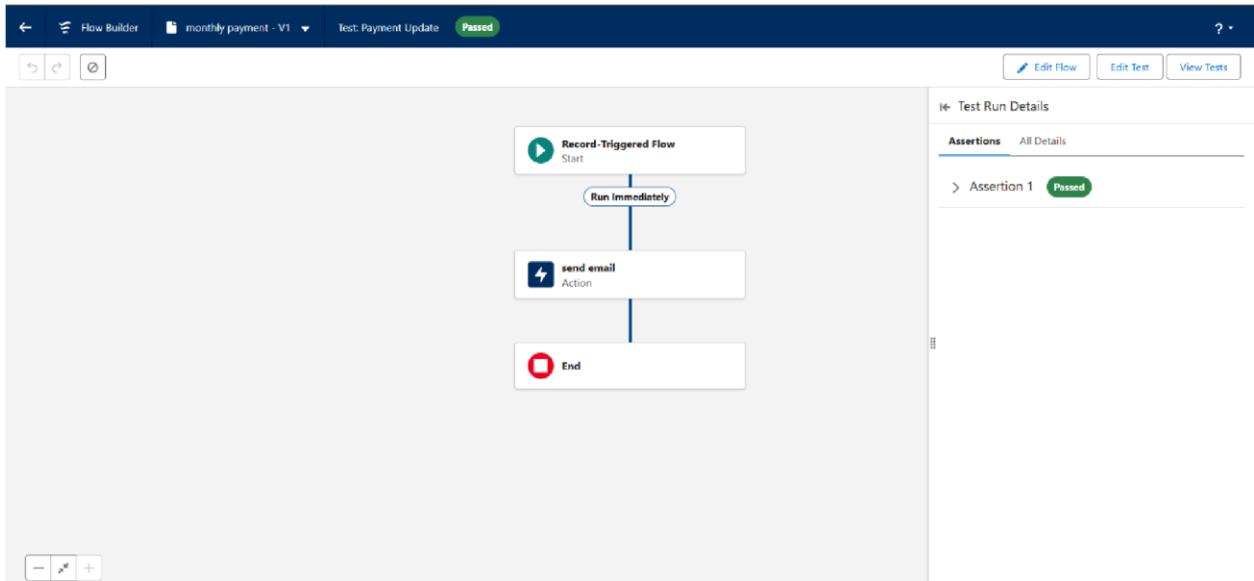


- Validation Rule checking



The screenshot shows a Salesforce Lease Management page for a lease named 'supriya'. The start date is set to 6/20/2025 and the end date to 6/19/2025. A validation error message 'We hit a snag.' appears, stating 'Review the following fields' and pointing to the start date field. The lease is owned by the 'Sowmya Team'.

- Test flows on payment update



- Approval process validated through email alerts and status updates

The screenshot shows a CRM interface for 'Lease Management' under the 'Tenants' tab. A search bar at the top right contains the text 'niranjan Approval'. The main area displays a 'Tenant' record for 'niranjan' with fields for Name, Email, Phone, Status (Stay), and Property (Parkside Lofts). The 'Owner' is listed as 'Sowmya Team'. Below the form are buttons for 'Cancel' and 'Save'. To the right, a sidebar titled 'Notifications' lists several recent messages:

- 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)

This screenshot shows the same CRM interface for 'Lease Management'. The 'Tenants' tab is selected, and the search bar again shows 'niranjan Approval'. The main area displays the same tenant record for 'niranjan'. To the right, a sidebar titled 'New Contact' has buttons for 'Edit' and 'New Opportunity'. Below this, a message states 'Get started by sending an email, scheduling a task, and more.' and 'No past activity. Past meetings and tasks marked as done show up here.' At the bottom, there is a section for 'Payment' with two entries: 'Jack' and 'Rahul', each with a 'View All' button.

RESULTS

Output Screenshots

- Tabs for Property, Tenant, Lease, Payment

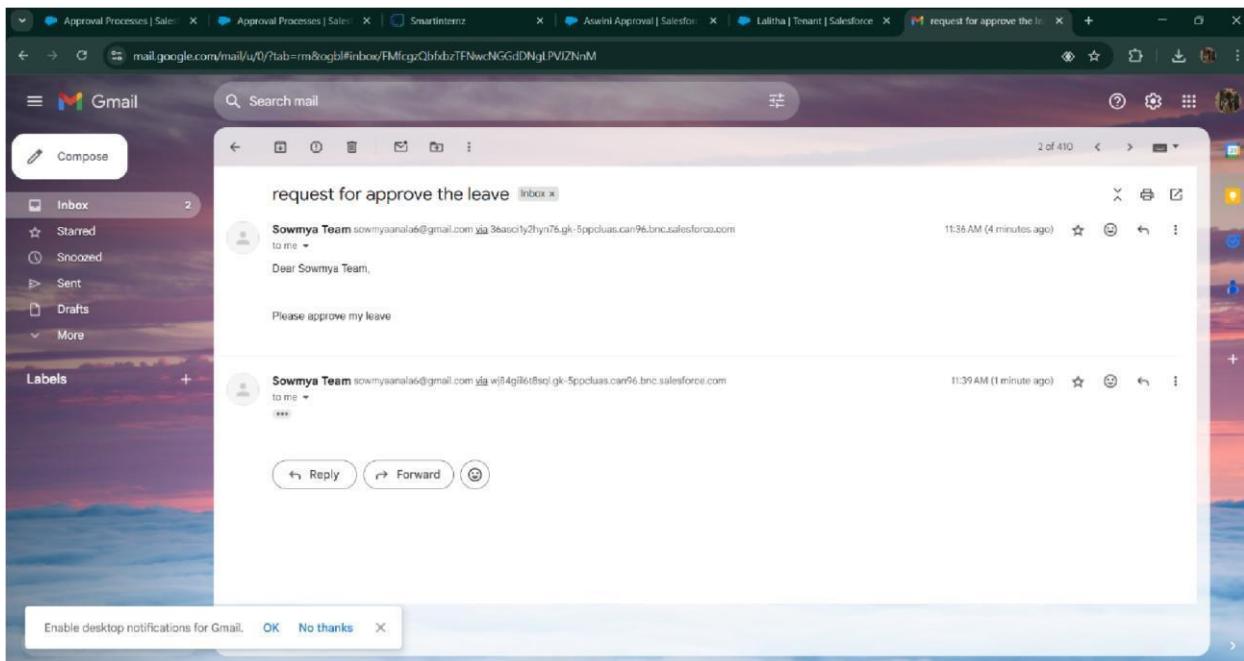
The screenshot shows the Salesforce Setup interface under the 'Tabs' section. It displays four custom tabs: 'Lease' (Tab Style: Keys), 'Payment' (Tab Style: Credit card), 'property' (Tab Style: Sack), and 'Tenant' (Tab Style: Map). The 'Custom Object Tabs' section also lists these tabs with their respective actions (Edit | Del) and labels.

- Email alerts

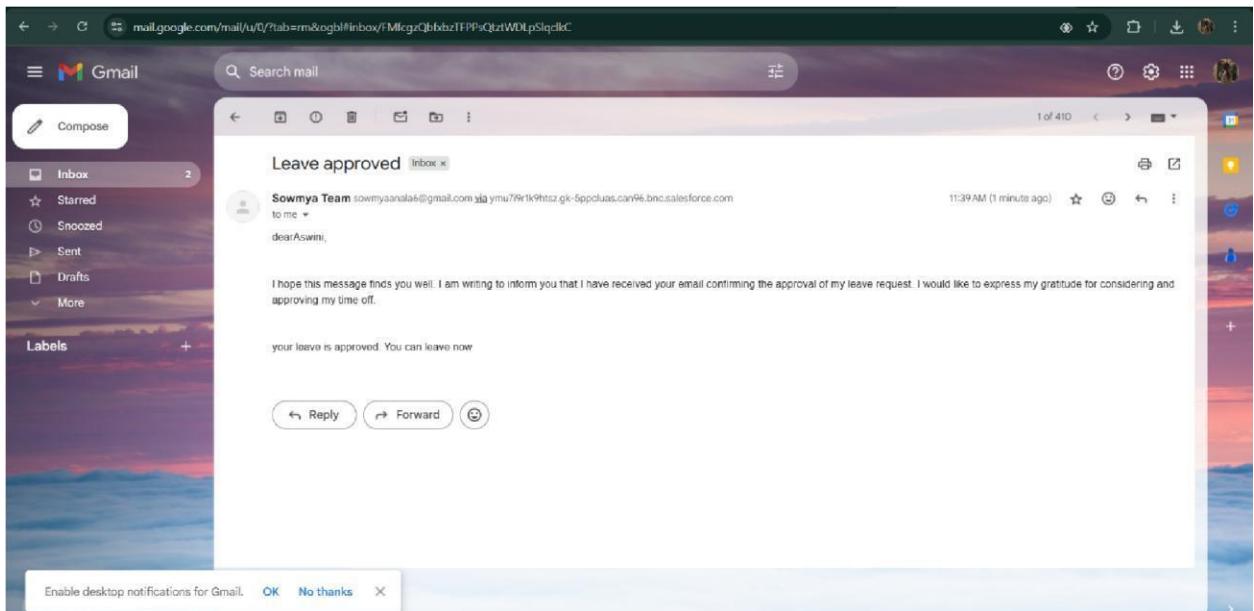
The screenshot shows the 'Approval History' list within the 'Lease Management' application. The table displays 8 items, sorted by 'Is Pending' and updated a few seconds ago. The columns include Step Name, Date, Status, Assigned To, Actual Approver, and Comments. The data shows a sequence of approval steps between June 23rd and 25th, 2025, involving the 'Sowmya Team'.

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

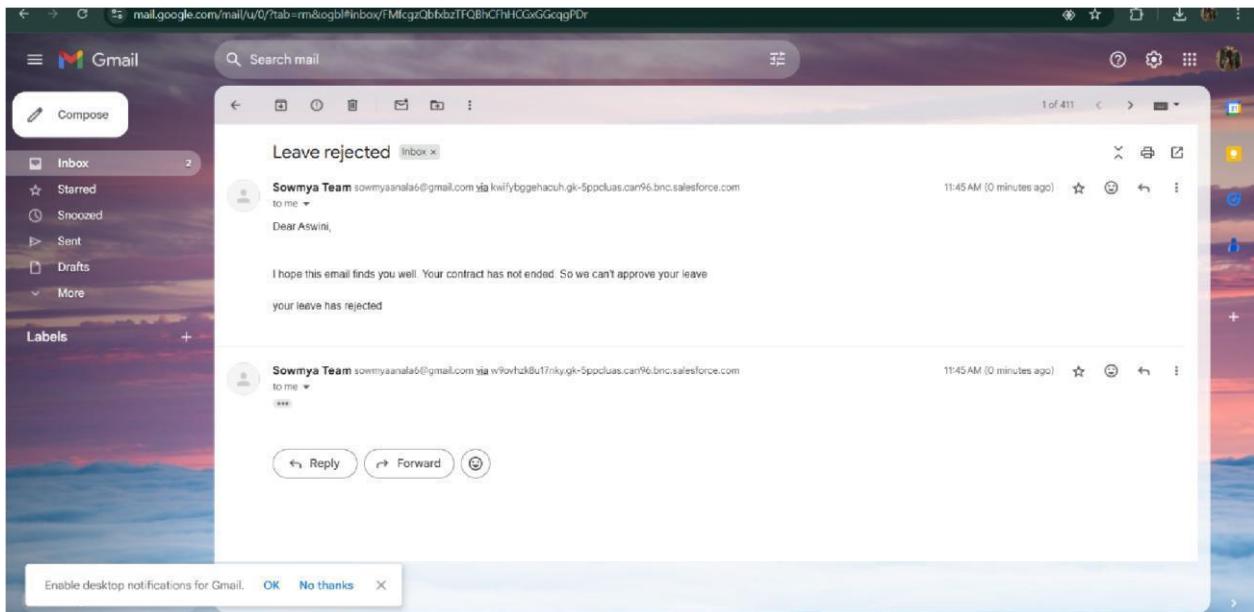
- Request for approve the leave



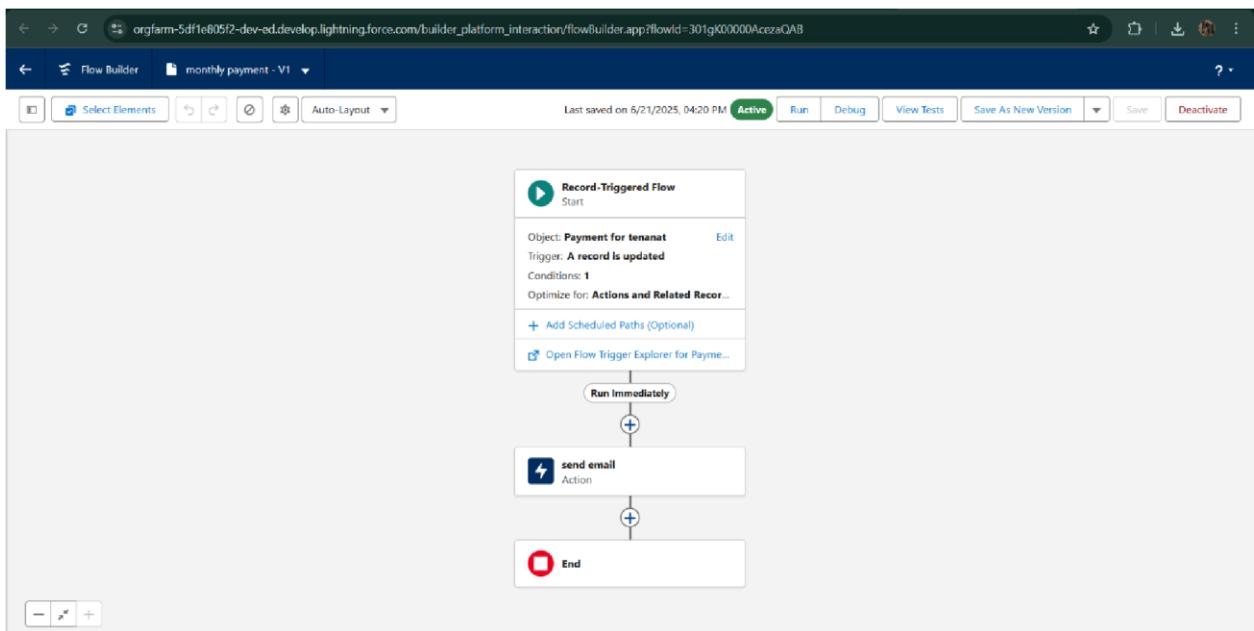
- Leave approved



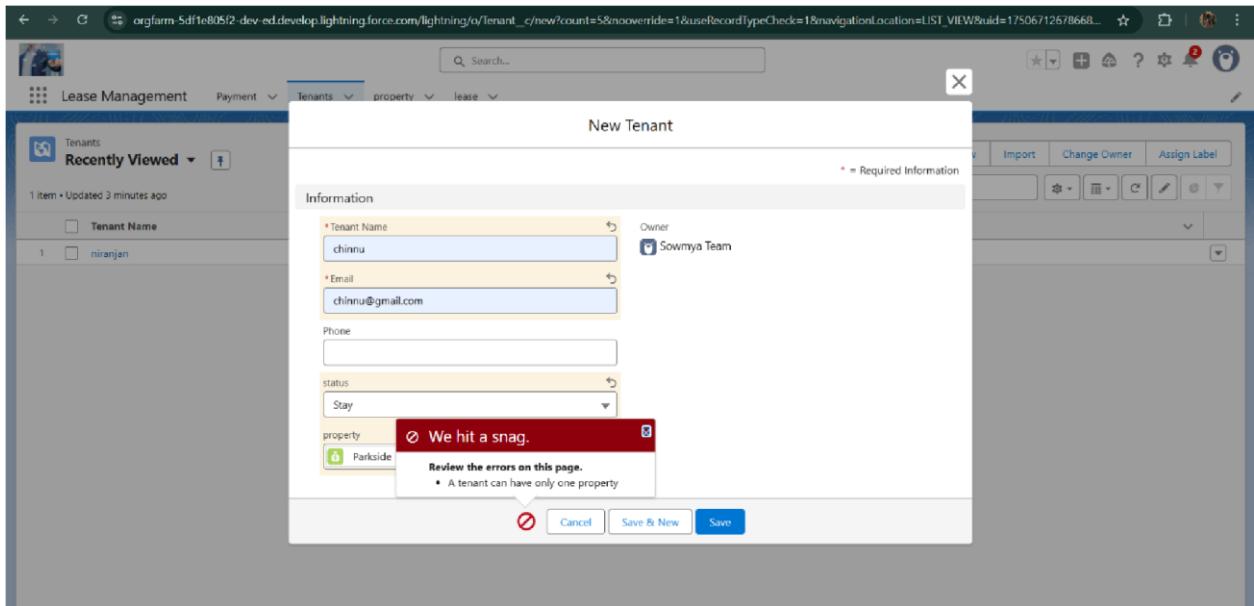
- Leave rejected



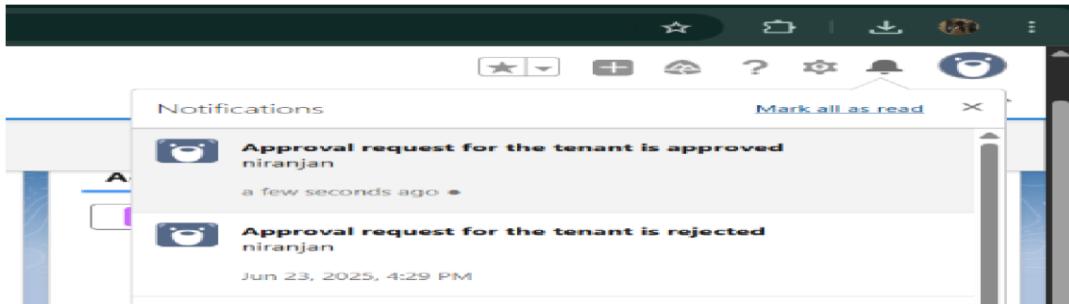
- Flow runs



- 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});  
  
    }  
  
}
```