

LEASE MANAGEMENT

College Name: KG College of Arts and Science

TEAM ID: NM2025TMID23743

TEAM MEMBERS:

Team Leader Name: Niroopa J N

Email: 2326ka29@kgcas.com

Team Member1: Nithish Kumar L

Email: 2326ka30@kgcas.com

Team Member: Nivitha K

Email: 2326ka31@kgcas.com

Team Member: Parama Dharshini H

Email: 2326ka32@kgcas.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.



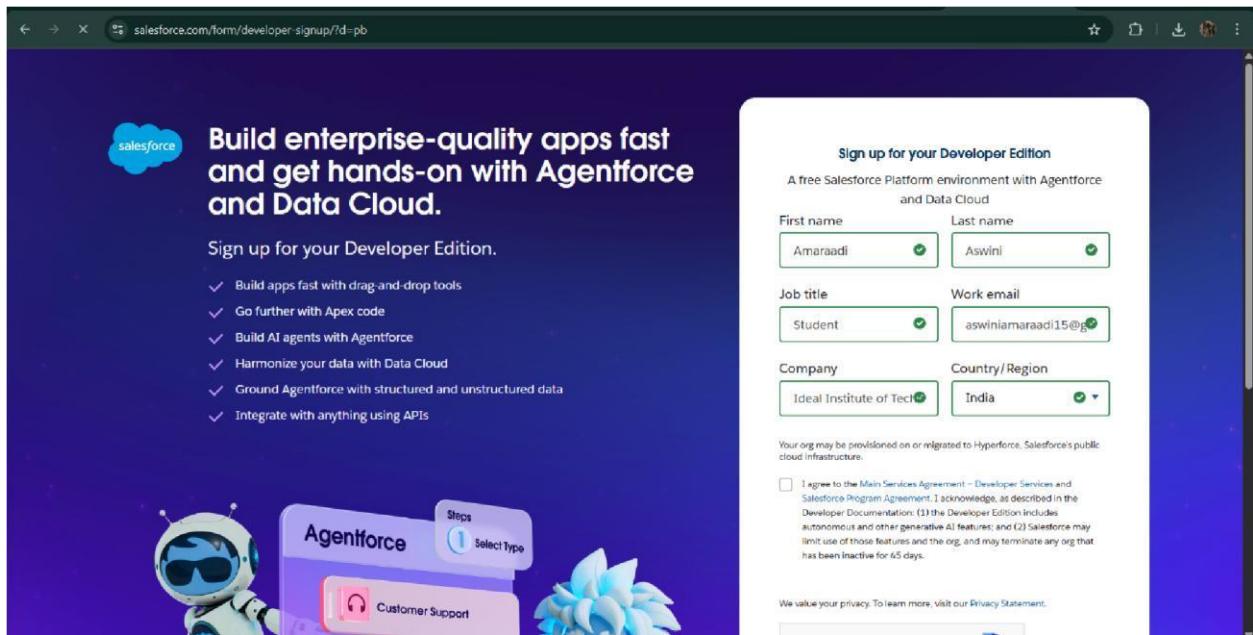
1.2 Purpose

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

orgfarm-5df1e805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK000000zTbN/Details/view

The screenshot shows the Salesforce Object Manager interface for the 'Tenant' object. The left sidebar lists various configuration tabs: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Restriction Rules, and Scoping Rules. The main 'Details' tab is selected. The 'Fields & Relationships' section contains fields for API Name (set to 'Tenant__c'), Singular Label (set to 'Tenant'), and Plural Label (set to 'Tenants'). The 'Buttons, Links, and Actions' section includes options for Enable Reports, Track Activities, Track Field History, Deployment Status (set to 'Deployed'), and Help Settings (set to 'Standard salesforce.com Help Window').

orgfarm-5df1e805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK000000zTuJ/Details/view

The screenshot shows the Salesforce Object Manager interface for the 'lease' object. The left sidebar lists the same configuration tabs as the Tenant object. The main 'Details' tab is selected. The 'Fields & Relationships' section contains fields for API Name (set to 'lease__c'), Singular Label (set to 'lease'), and Plural Label (set to 'lease'). The 'Buttons, Links, and Actions' section includes options for Enable Reports, Track Activities, Track Field History, Deployment Status (set to 'Deployed'), and Help Settings (set to 'Standard salesforce.com Help Window').

The screenshot shows the Salesforce Setup interface under 'Object Manager'. A sidebar on the left lists various configuration options like Details, Fields & Relationships, Page Layouts, etc. The main area displays the 'Details' tab for the 'Payment for tenant' object. The 'Description' section includes fields for API Name ('Payment_for_tenant__c'), Singular Label ('Payment for tenant'), and Plural Label ('Payment'). The 'Enable Reports' section contains checkboxes for Track Activities, Track Field History, Deployment Status, and Help Settings, all of which are checked. At the top right are 'Edit' and 'Delete' buttons.

- Configured fields and relationships

The screenshot shows the 'Fields & Relationships' section for the 'property' object. The sidebar on the left shows the 'Fields & Relationships' tab is selected. The main table lists the following fields:

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

Setup > Object Manager
lease

Fields & Relationships					
FIELD LABEL		FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By		CreatedById	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		

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: 

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



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

App Settings

App Details & Branding
App Options
Utility Items (Desktop Only)

Navigation Items

User Profiles

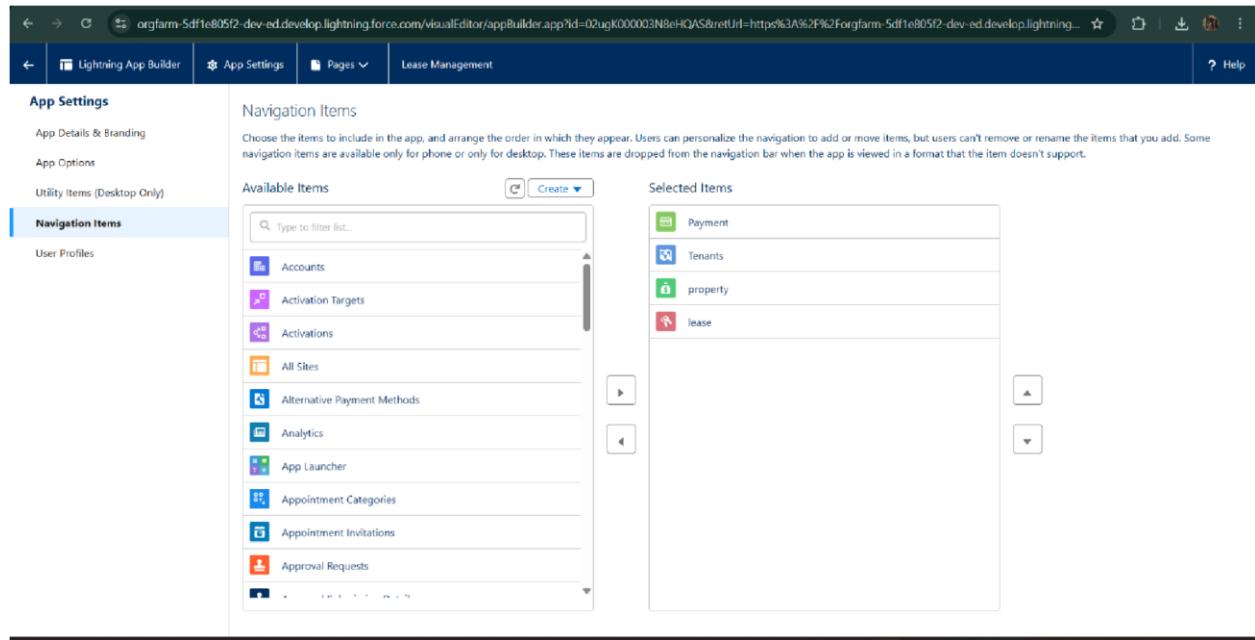
Available Items

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 | Help

App Settings

App Details & Branding
App Options
Utility Items (Desktop Only)

Navigation Items

User Profiles

Choose the user profiles that can access this app.

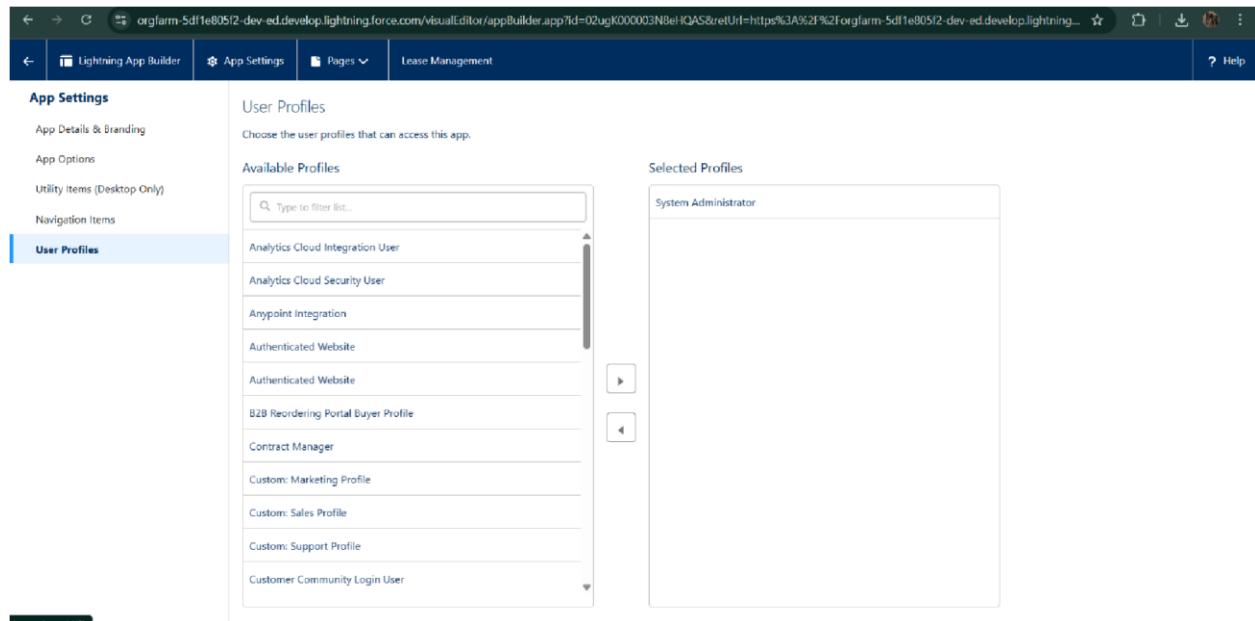
Available Profiles

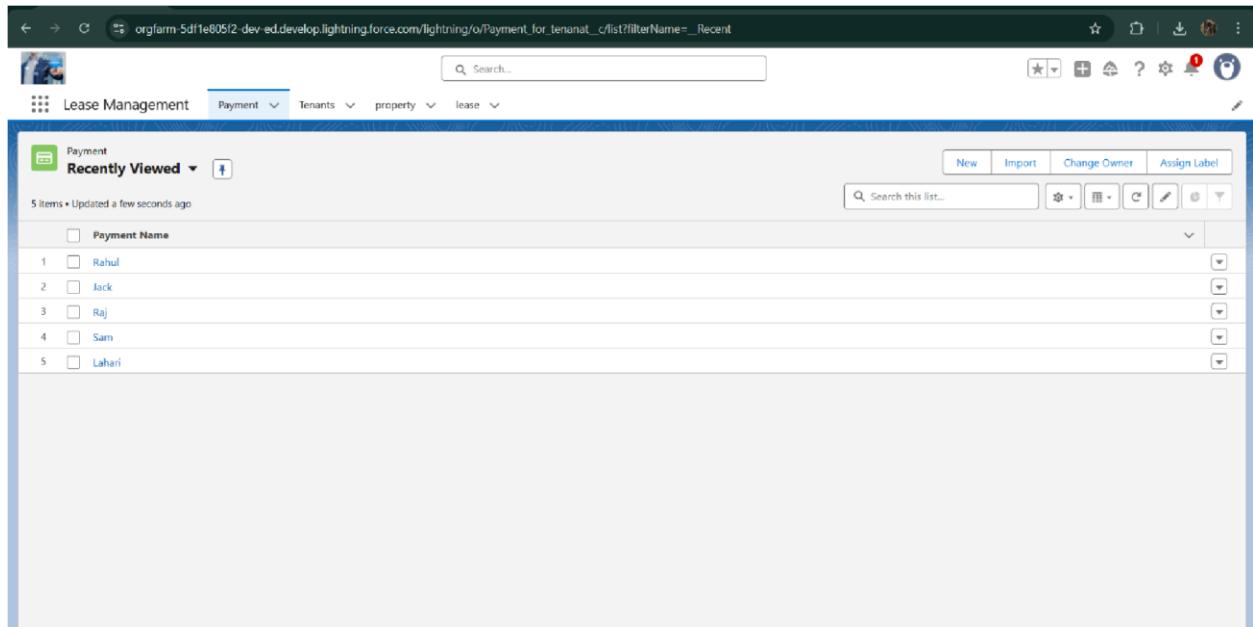
Type to filter list...

- 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

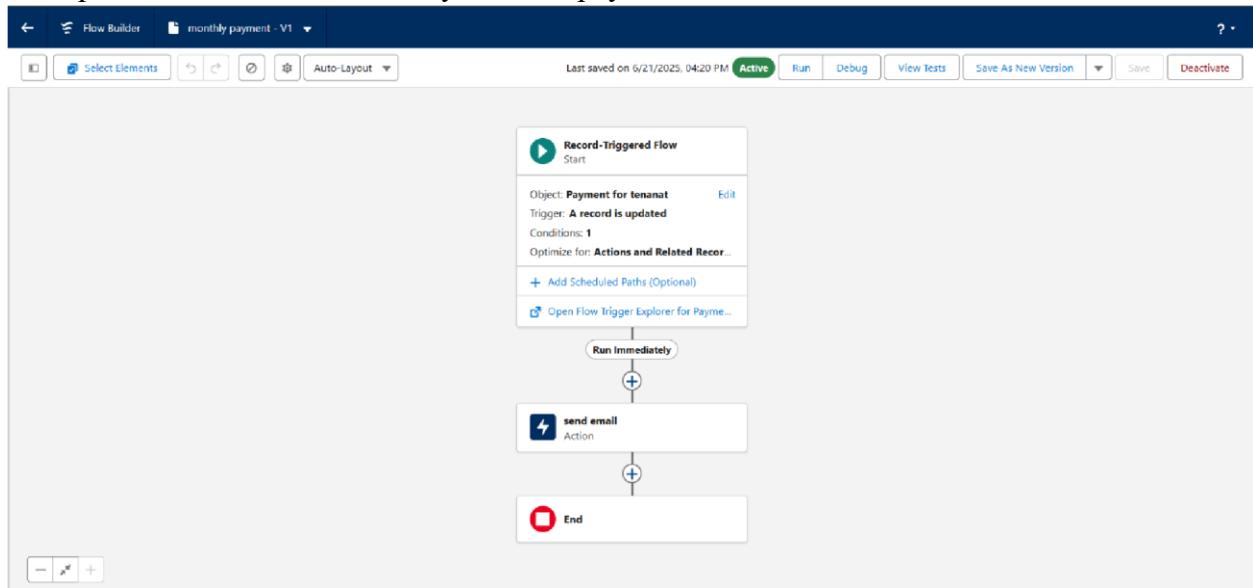
Selected Profiles

- 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

`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

Check Syntax

Error Messages

SETUP > OBJECT MANAGER
lease

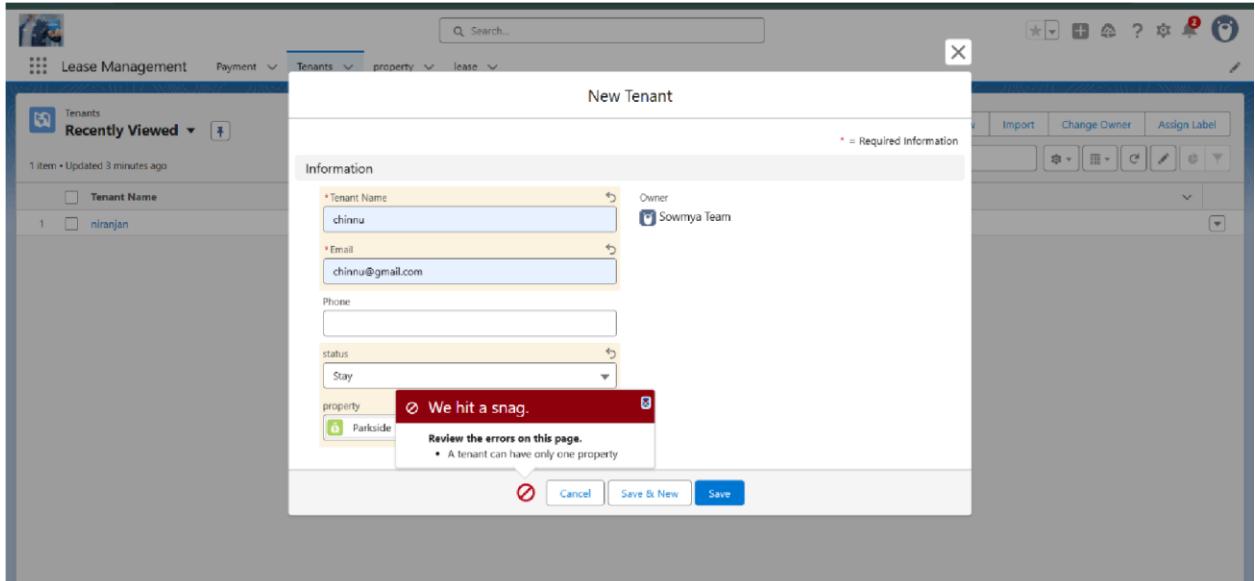
lease Validation Rule

[Help for this Page](#)

Validation Rule Detail

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

- Added Apex trigger to restrict multiple tenants per property



- Scheduled monthly reminder emails using Apex class

The screenshot shows a Java code editor with the following details:

- Title Bar:** File, Edit, Debug, Test, Workspaces, Help.
- Toolbar:** New, Open, Save, Run, Stop, Refresh, Code Coverage, New API, Version 44.
- Code Area:** The code is for a class named `MonthlyEmailScheduler` which implements the `Schedulable` interface. The code checks if it's the first day of the month and sends monthly emails to tenants.
- Code Content:**

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

- Bottom Navigation:** Help, Tools, Checkpoints, Query Editor, View Source, Progress, Problems.
- Status Bar:** Line 44, Column 1.

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

The screenshot shows the Salesforce Setup interface with the 'Email' section selected. Under 'Classic Email Templates', the 'Leave approved' template is displayed. The template details are as follows:

Email Template Detail

- Email Template Name: Leave approved
- Template Unique Name: Leave_approved
- Encoding: Unicode (UTF-8)
- Author: Sowmya Team [Change]
- Description: Created By: Sowmya Team, 6/20/2025, 1:08 AM
- Modified By: Sowmya Team, 6/20/2025, 1:08 AM
- Available For Use: ✓
- Last Used Date: Times Used: 0

Email Template

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 'Email' section selected. Under 'Classic Email Templates', the 'tenant leaving' template is displayed. The template details are as follows:

Email Template Detail

- Email Template Name: tenant leaving
- Template Unique Name: tenant_leaving
- Encoding: Unicode (UTF-8)
- Author: Sowmya Team [Change]
- Description: Created By: Sowmya Team, 6/20/2025, 1:06 AM
- Modified By: Sowmya Team, 6/20/2025, 1:06 AM
- Available For Use: ✓
- Last Used Date: Times Used: 0

Email Template

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 following details:

Classic Email Templates

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	Soumya_Team [Change]
Description	
Created By	Soumya_Team 6/20/2025, 1:11 AM
Modified By	Soumya_Team 6/20/2025, 1:11 AM

Email Template

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
Template Unique Name	Tenant_Email
Encoding	Unicode (UTF-8)
Author	Soumya_Team [Change]
Description	
Created By	Soumya_Team 6/20/2025, 1:12 AM
Modified By	Soumya_Team 6/20/2025, 1:12 AM

Email Template

Subject: Urgent: Monthly Rent Payment Reminder

Plain Text Preview

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

The screenshot shows the Salesforce Setup interface with the 'Email' section selected. Under 'Email', 'Classic Email Templates' is chosen. A search bar at the top left shows 'email template'. The main content area displays a 'Text Email Template' named 'tenant payment'. The 'Email Template Detail' section shows the following information:

- Email Templates from Salesforce: Unfiled Public Classic Email Templates
- Email Template Name: tenant payment
- Template Unique Name: tenant_payment
- Encoding: Unicode (UTF-8)
- Author: Scamya Team [Change]
- Description: Created By: Scamya Team, 6/20/2025, 1:13 AM
- Available For Use: checked
- Last Used Date: Times Used: 0
- Modified By: Scamya Team, 6/20/2025, 1:13 AM

The 'Email Template' section below shows the subject 'Confirmation of Successful Monthly Payment' and a plain text preview:

```

Subject : Confirmation of Successful Monthly Payment
Plain Text Preview
Dear {Tenant__c_Email__c}.

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

```

● Approval Process creation

For Tenant Leaving:

The screenshot shows the Salesforce Setup interface with the 'Data' section selected. Under 'Data', 'Approval Processes' is chosen. A search bar at the top left shows 'approval'. The main content area displays an 'Approval Process' named 'Tenant: TenantApproval'. The 'Process Definition Detail' section shows the following information:

- Process Name: TenantApproval
- Unique Name: TenantApproval
- Description:
- Entry Criteria: tenant__c.status EQUALS Slay
- Record Editability: Administrator ONLY
- Active: checked
- Next Automated Approver Determined By:
- Allow Submitters to Recall Approval Requests: unchecked
- Created By: Scamya Team, 6/23/2025, 3:41 AM
- Modified By: Scamya Team, 6/26/2025, 11:57 PM

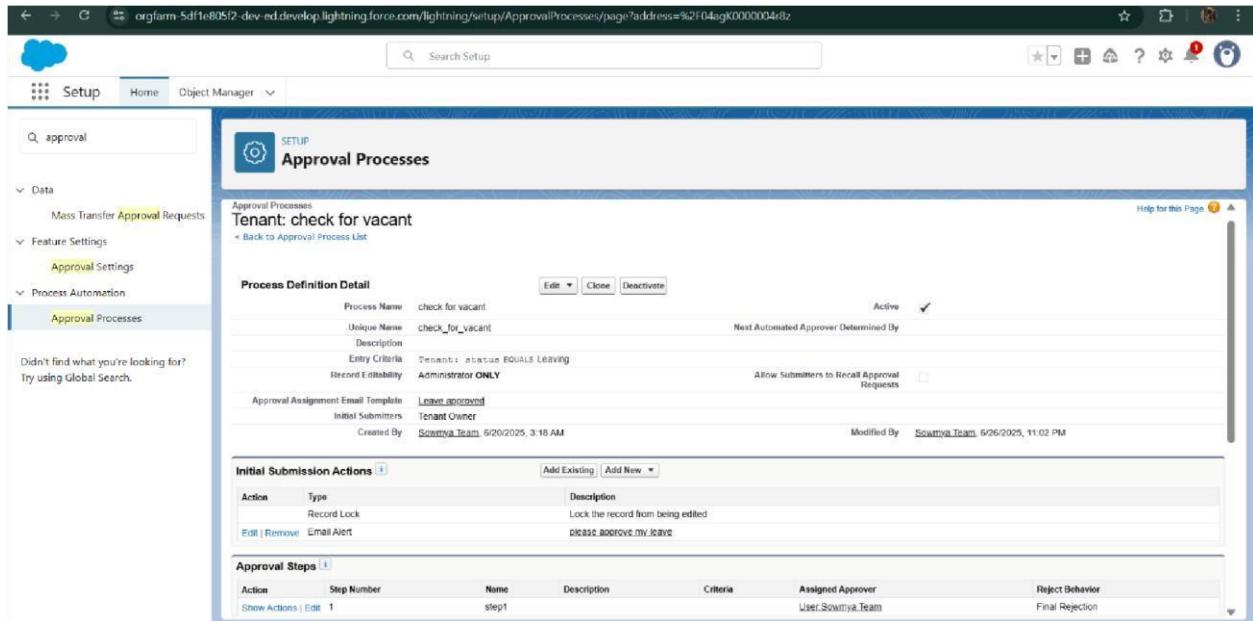
The 'Initial Submission Actions' section shows:

Action Type	Description
Record Lock	Lock the record from being edited

The 'Approval Steps' section shows:

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

For Check for Vacant:



● Apex Trigger

Create an Apex Trigger

The screenshot shows the Eclipse IDE with an Apex trigger named 'testHandler' open. The code is as follows:

```

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

```

A tooltip window is displayed over the 'testHandler' class name, listing available entities:

Entity Type	Entities	Related
Entity Type	Name Namespace	Name Extent Direction
Classes	test	
Triggers		
Pages		
Page Components		
Objects		
Static Resources		
Packages		

```

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

```

Create an Apex Handler class

```

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');
            }
        }
    }
}

```

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is https://orgfarm-5df1e805f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The tab bar has 'testHandler.apc' selected. The code editor contains the following Apex class:

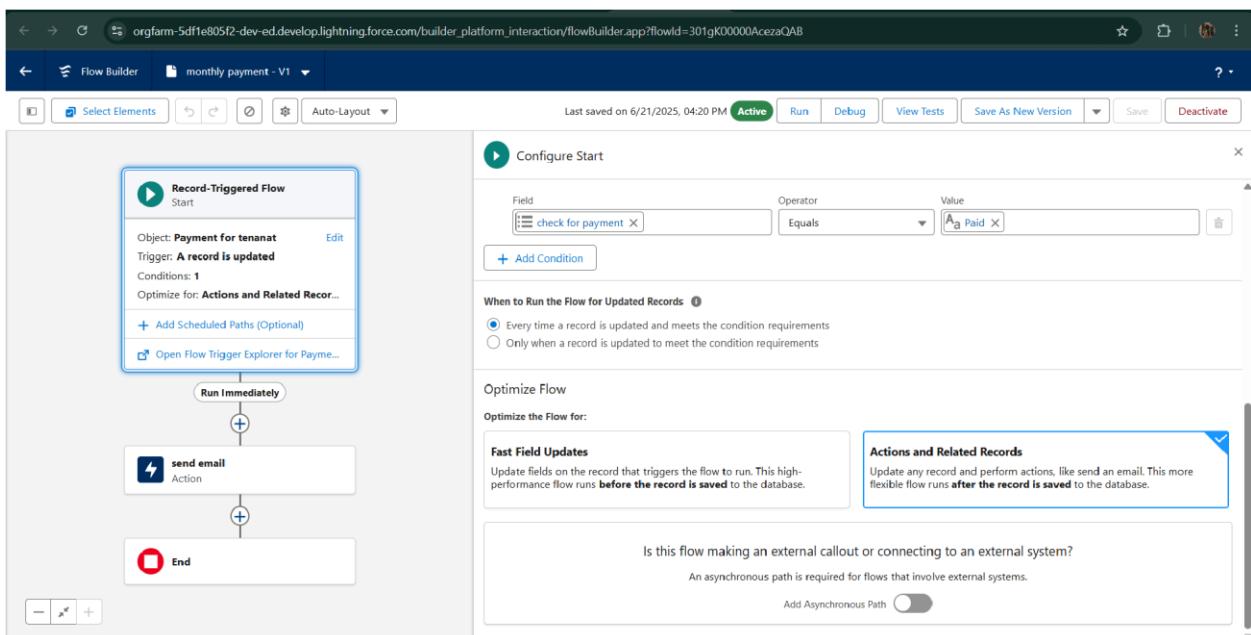
```

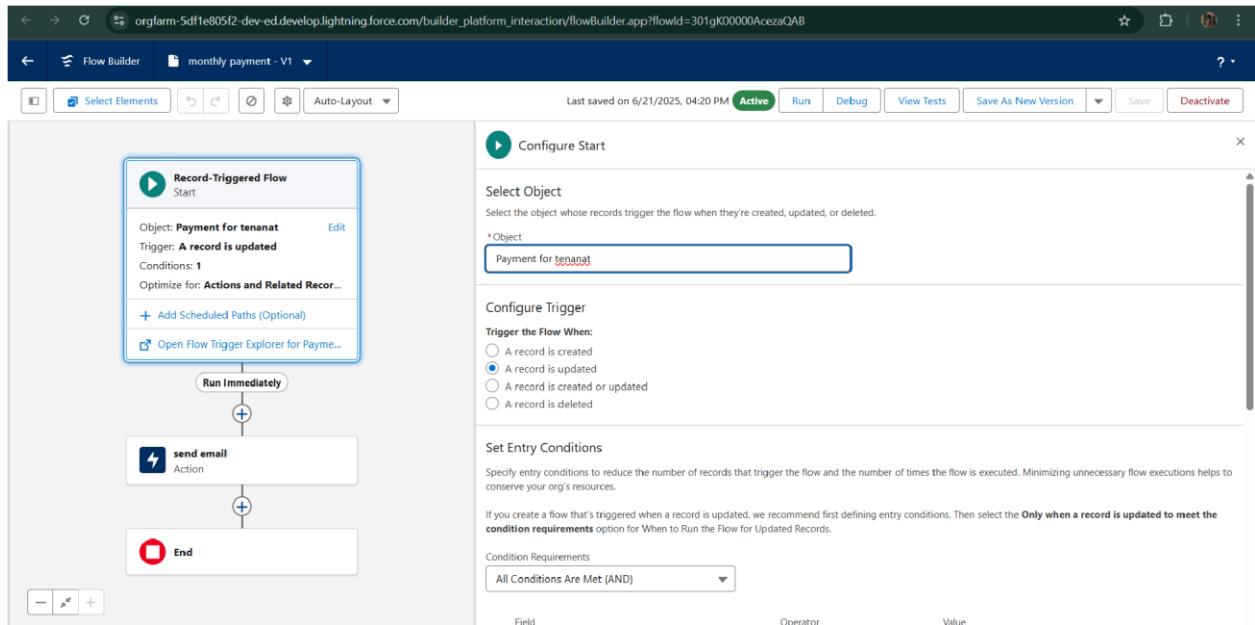
1 * public class testHandler {
2
3     public static void preventInsert(List<Tenant__c> newList) {
4
5         Set<Id> existingPropertyIds = new Set<Id>();
6
7         for (Tenant__c existingTenant : [SELECT Id, Property__c FROM Tenant__c WHERE Property__c != null]) {
8
9             existingPropertyIds.add(existingTenant.Property__c);
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 }

```

The status bar at the bottom shows 'Logs Tests Checkpoints Query Editor View State Progress Problems'. The 'Problems' tab is selected.

● 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 sendMonthlyEmails()
17
18     List<Tenant__c> tenants = [SELECT
19
20         for (Tenant__c tenant : tenants
21
22             String recipientEmail = tenant.Email__c;
23

```

Open

Entity Type	Name	Namespace	Related
Classes	testHandler		Email CustomField Object Tenant__c Subject
Triggers	MonthlyEmailScheduler		
Pages			
Page Components			
Objects			
Static Resources			
Packages			

```

1. *@ISOBILL 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 = 'Wesider: monthly rent payment due';
27.
28.            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
29.
30.            email.setToAddresses(new String[]{recipientEmail});
31.
32.            email.setSubject(emailSubject);
33.
34.            email.setPlainTextBody(emailContent);
35.
36.            Messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
37.
38.        }
39.
40.    }
41.
42. }

```

Schedule Apex class

The screenshot shows the Salesforce Setup interface. The left sidebar has a search bar with 'apex' typed in. Under 'Apex Classes', the 'MonthlyEmailScheduler' class is selected. The main area displays the Apex code for the class.

```

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

```

Lease Management Payment Tenants property lease

Tenant: Aswini

Related Details * = Required Information

Tenant Name: Aswini Owner: Sowmya Team

Email: aswiniamaraadi15@gmail.com

Phone: (905) 223-5567

Status: Leaving

Property: Imran

Created By: Sowmya Team, 6/26/2025, 6:05 AM Last Modified By: Sowmya Team, 6/26/2025, 11:06 PM

New Contact Edit New Opportunity

Activity

New Case New Lead Delete Clone Change Owner Refresh Printable View Submit for Approval Edit Labels

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.

Lease Management Payment Tenants property lease

Tenant: Aswini

Related Details * = Required Information

Tenant was submitted for approval.

Tenant Name: Aswini Owner: Sowmya Team

Email: aswiniamaraadi15@gmail.com

Phone: (905) 223-5567

Status: Leaving

Property: Imran

Created By: Sowmya Team, 6/26/2025, 6:05 AM Last Modified By: Sowmya Team, 6/26/2025, 11:06 PM

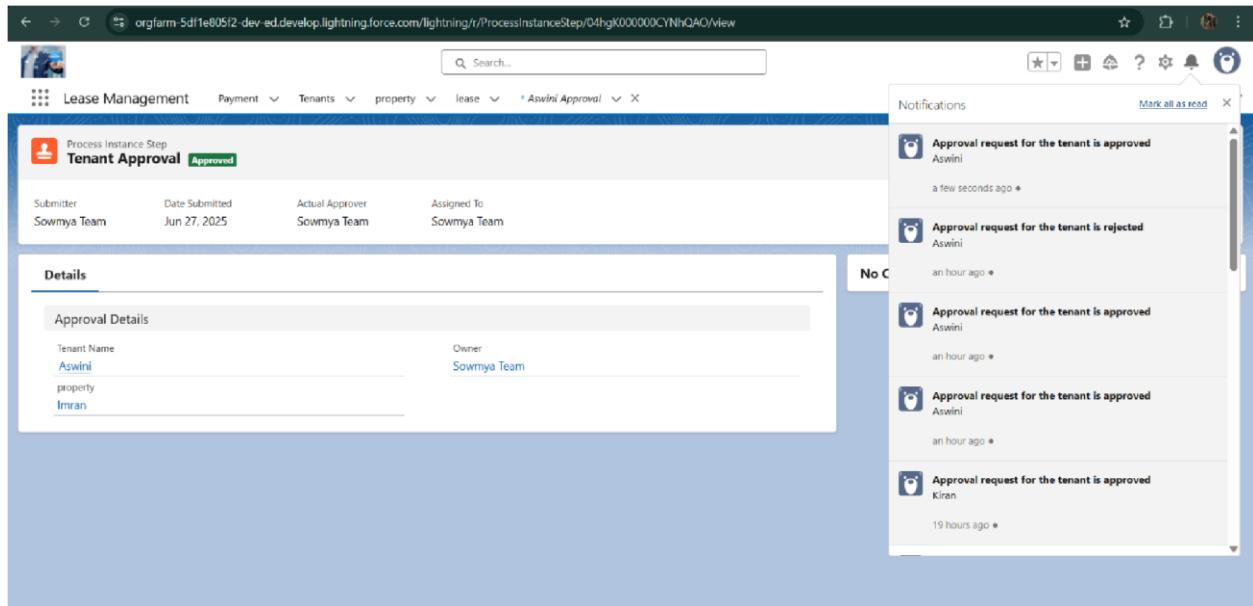
New Contact Edit New Opportunity

Activity

New Case New Lead Delete Clone Change Owner Refresh Expand All View All

Upcoming & Overdue No activities to show. Get started by sending an email, scheduling a task, and more.

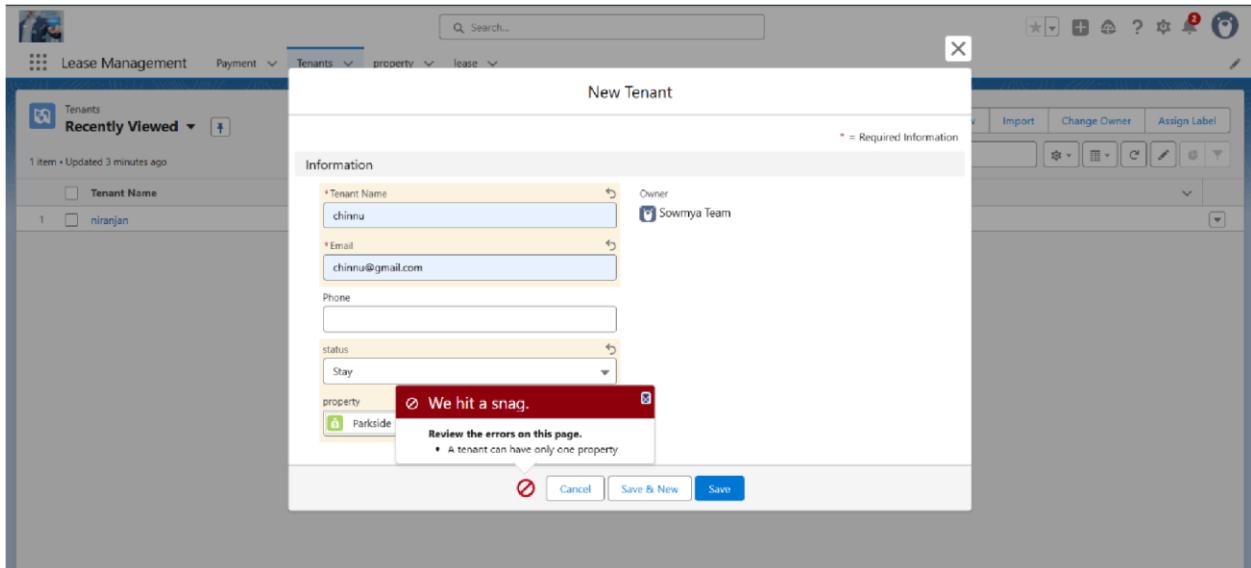
No past activity. Past meetings and tasks marked as done show up here.



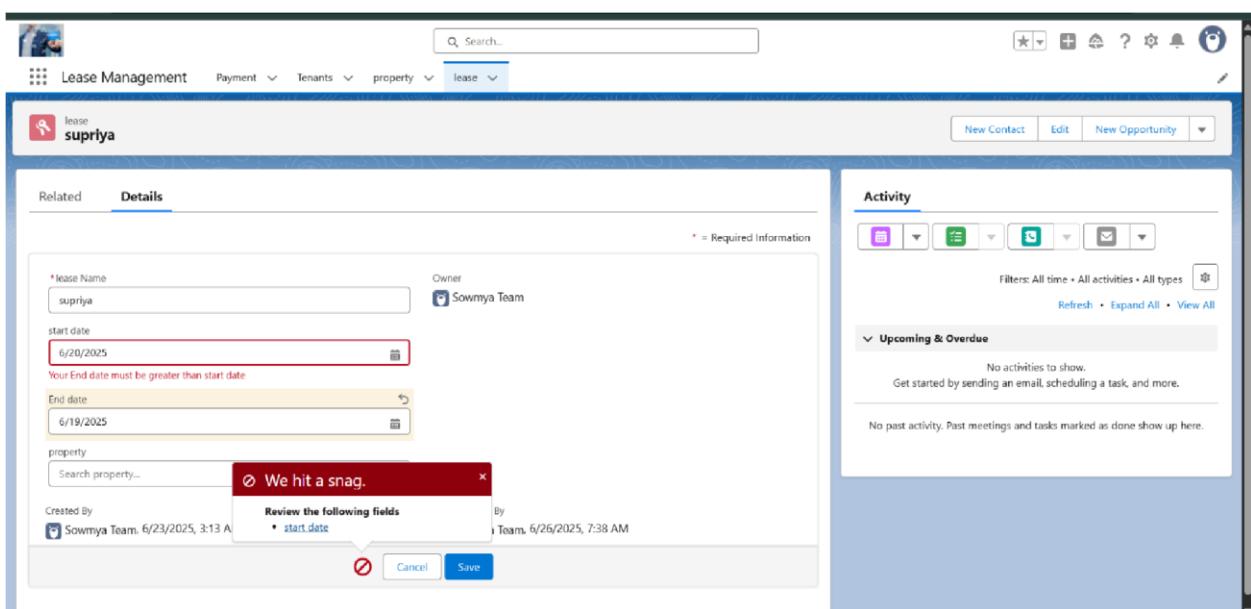
FUNCTIONAL AND PERFORMANCE TESTING

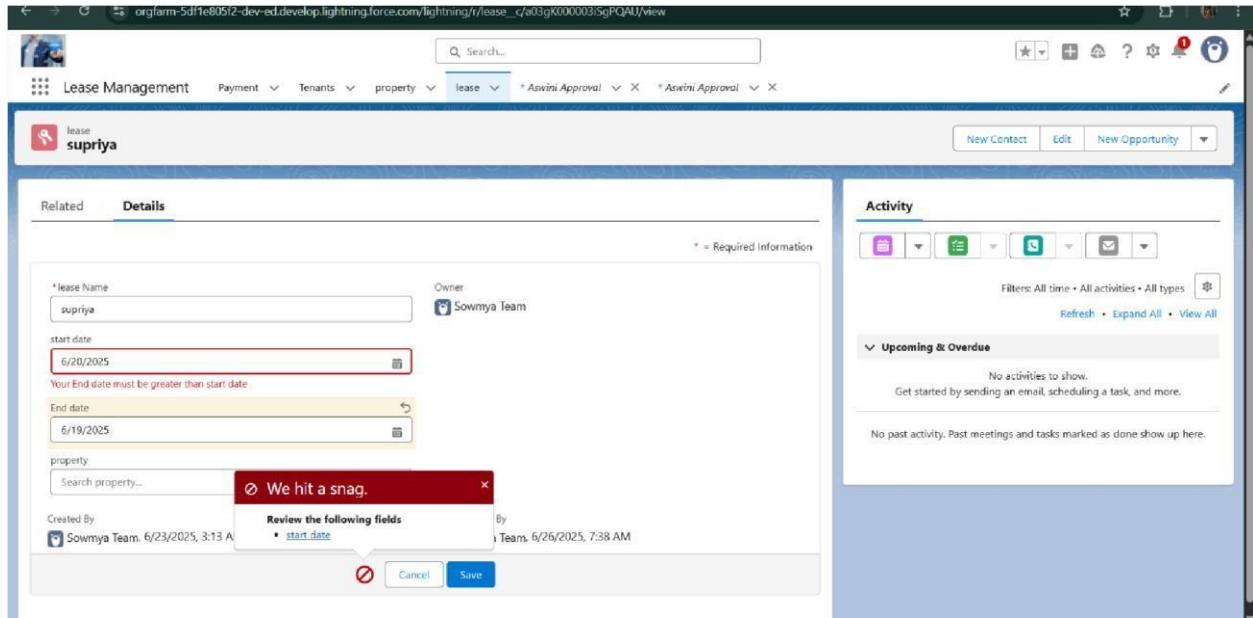
Performance Testing

- Trigger validation by entering duplicate tenant-property records

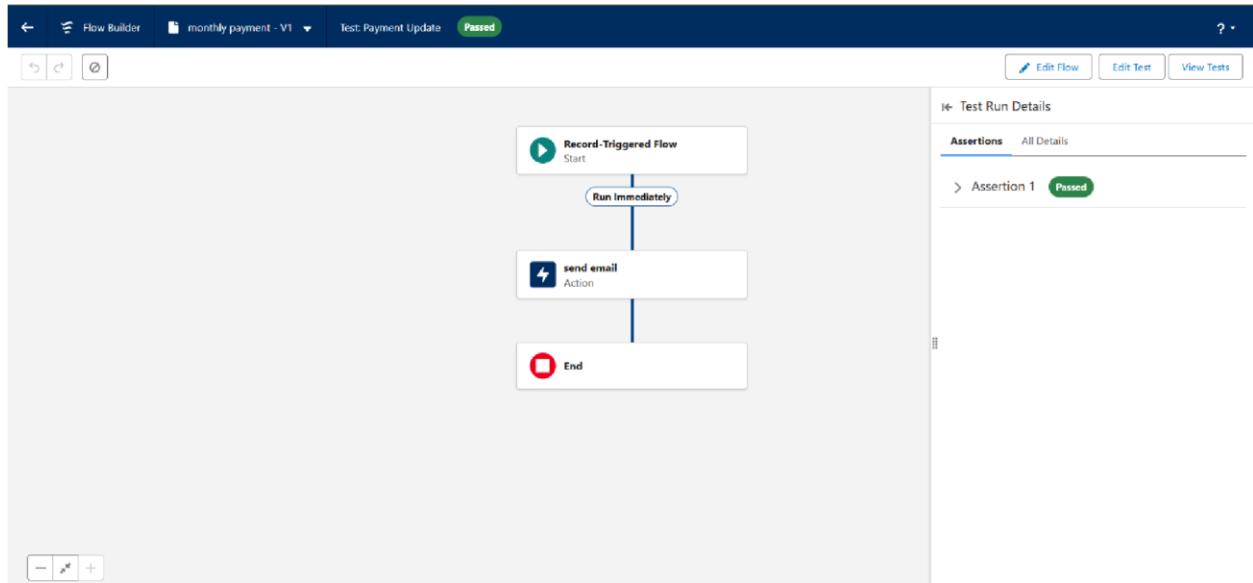


- Validation Rule checking





- Test flows on payment update



- Approval process validated through email alerts and status updates

The screenshot shows a tenant record for "niranjan". The "Details" tab is selected. The form includes fields for Tenant Name (niranjan), Email (niranjan1506@gmail.com), Phone, status (Stay), and property (Parksid Lofts). The "Owner" field is set to "Sowmya Team". The "Created By" field shows "Sowmya Team" on 6/23/2025, 2:33 AM, and the "Last Modified By" field shows "Sowmya Team" on 6/23/2025, 3:58 AM. A "Save" button is at the bottom.

Notifications

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

The screenshot shows the same tenant record for "niranjan". The "Approval History" section displays six entries:

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

The "Payment" section shows two entries for "Jack" and "Rahul".

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

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

RESULTS

Output Screenshots

- Tabs for Property, Tenant, Lease, Payment

The screenshot shows the Salesforce Setup interface under the 'Custom 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	Key	
Edit Del	Payment	Credit card	
Edit Del	property	Bank	
Edit Del	Tenants	Map	

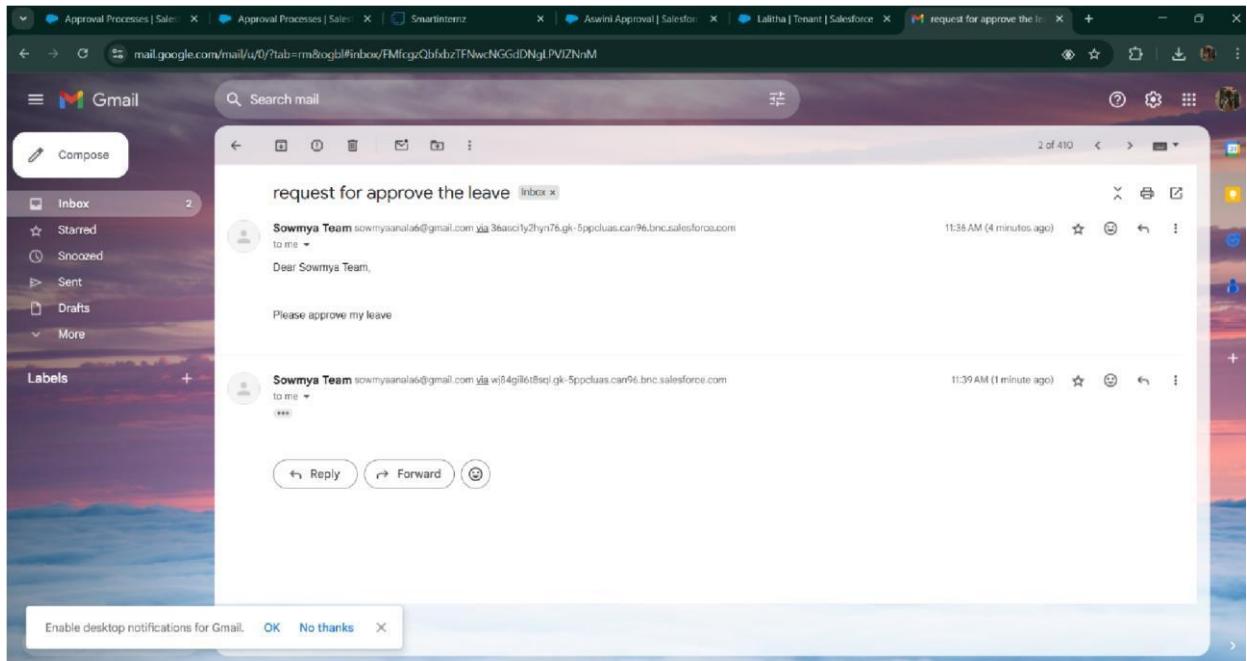
Below the table, sections for 'Web Tabs' and 'Visualforce Tabs' show that no tabs have been defined for those categories.

- Email alerts

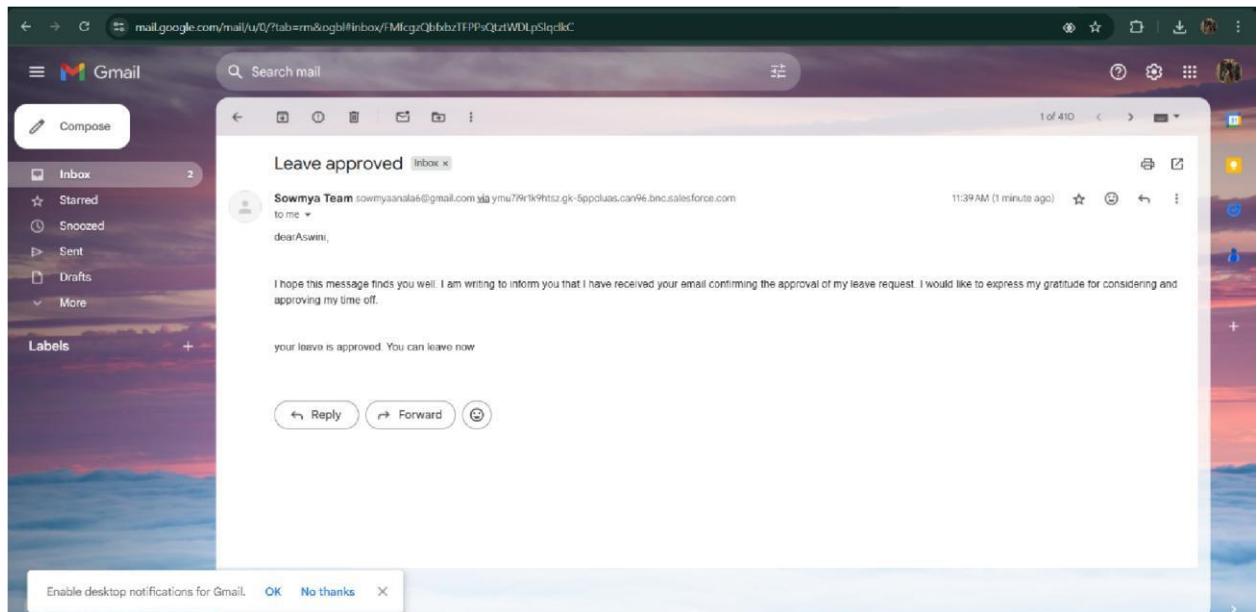
The screenshot shows the 'Lease Management' application interface. A table titled 'Approval History' lists 8 items, sorted by Is Pending, updated a few seconds ago. The columns include Step Name, Date, Status, Assigned To, Actual Approver, and Comments. The data is as follows:

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

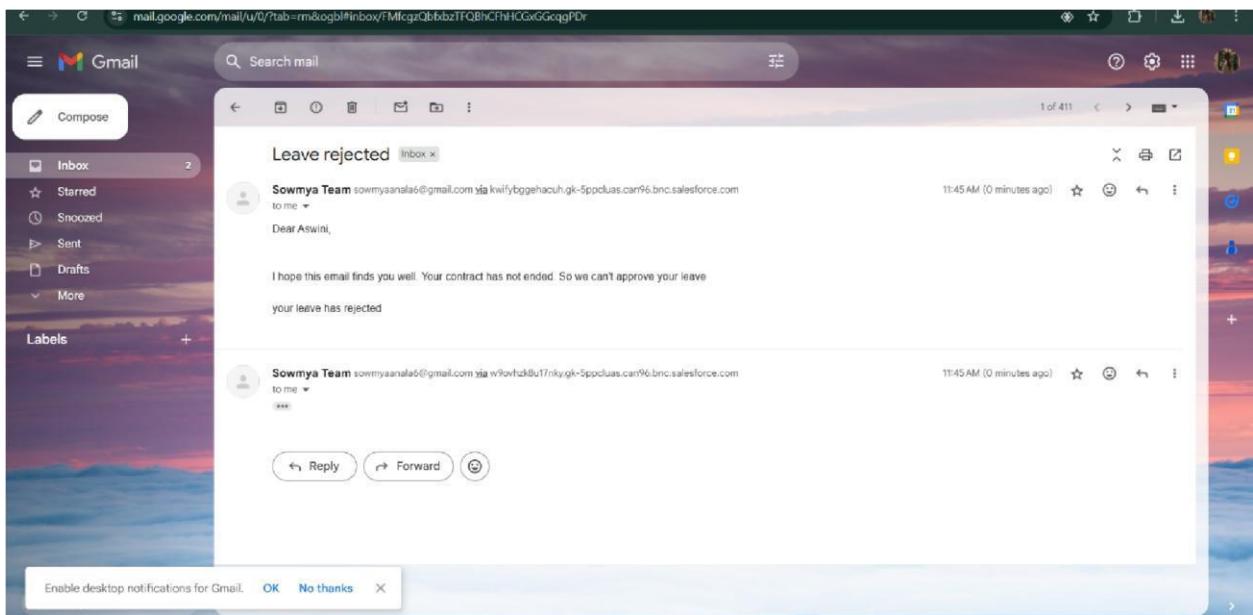
- Request for approve the leave



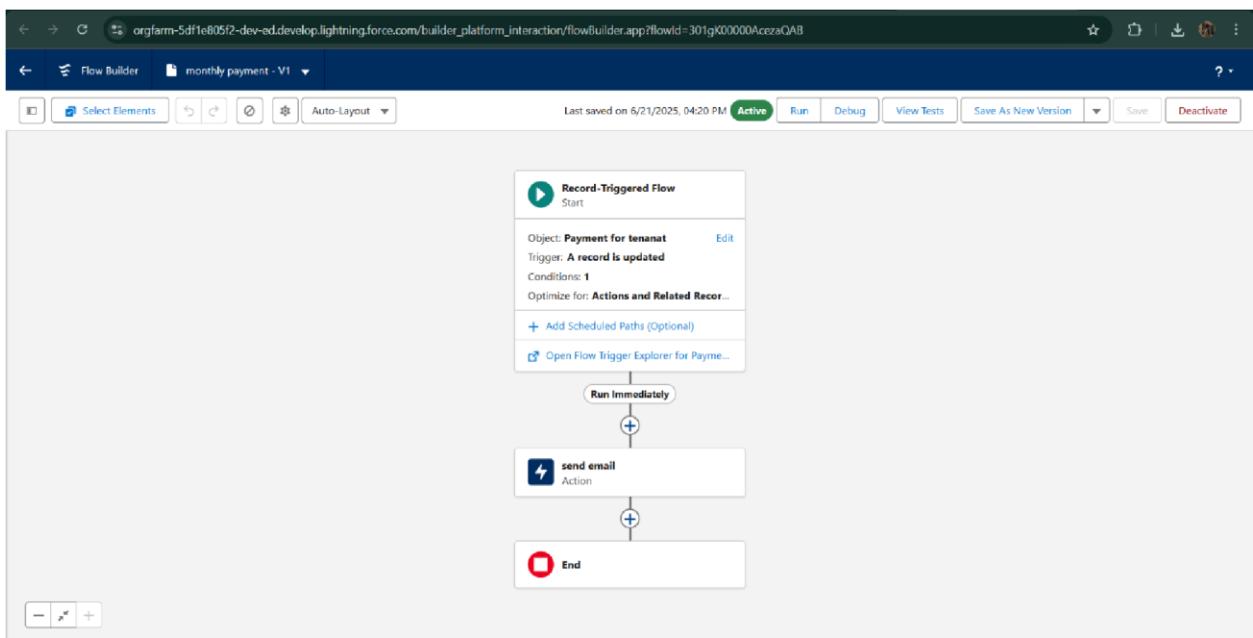
- Leave approved



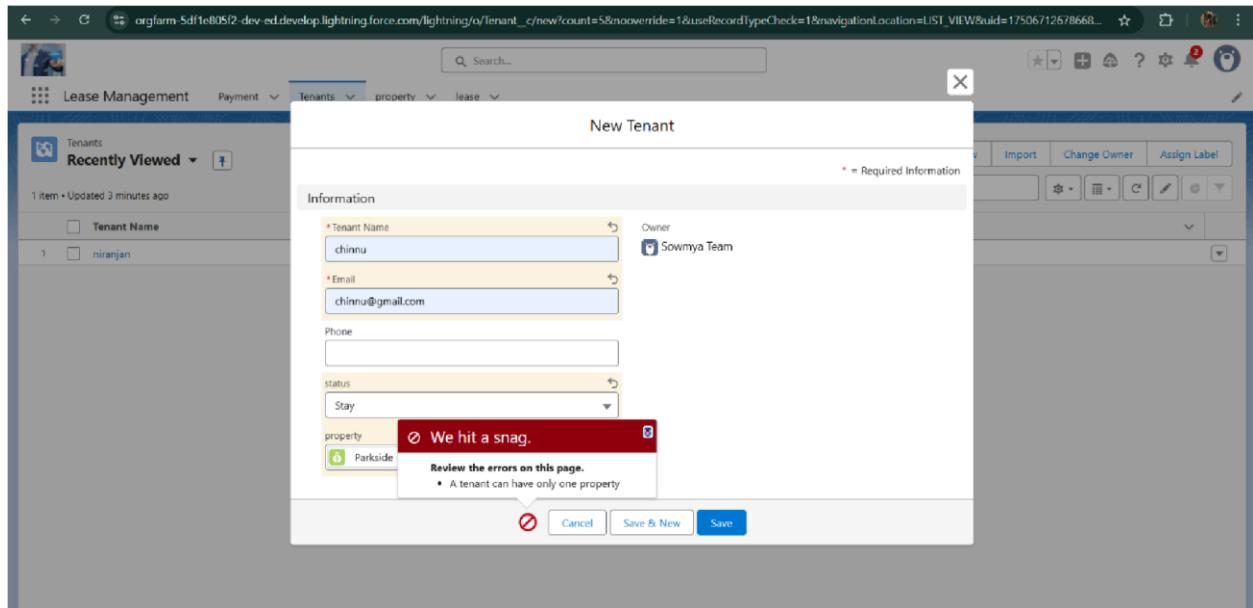
- Leave rejected



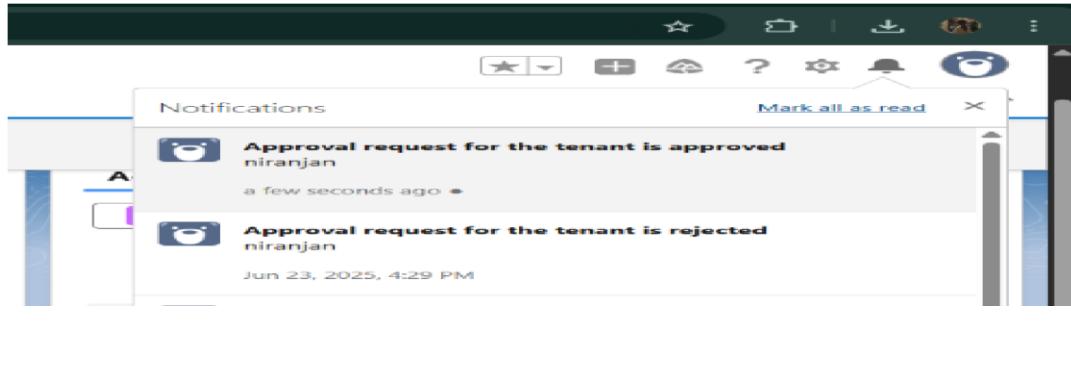
- Flow runs



- Trigger error messages



- Approval process notifications



ADVANTAGES & DISADVANTAGES

ADVANTAGE

Lower upfront cost
Increased flexibility

DISADVANTAGE

Higher Long – term costs
No ownership or equity

Reduced responsibility

Lack of control

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)) { newTenantaddError('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});

}
```