

LEASE MANAGEMENT

**College Name: Kumaraguru College of Liberal Arts And Science
College Code: BRUAX**

TEAM ID: NM2025TMID25588

TEAM MEMBERS: 4

Team Leader Name: MIDHUNESH G

Email: midhunesh.23bds@kclas.ac.in

Team Member1: AZWIN MUHAMMED KK

Email: azwinmuhammed.23bds@kclas.ac.in

Team Member: NITIN AADITHYA SUNIL KUMAR

Email: nitinaadithya.23bds@kclas.ac.in

Team Member: JISHNU J

Email: jishnu.23bds@kclas.ac.in

1. INTRODUCTION

1.1 Project Overview

The Lease Management System is a Salesforce-based application developed to simplify and streamline the end-to-end process of managing real estate leases. It helps organizations handle tenant information, lease agreements, payment schedules, and ongoing communications in one centralized platform. By leveraging automation features like flows, approval processes, and email alerts, the system reduces manual effort, speeds up operations, and improves overall efficiency.

1.2 Purpose

The purpose of this project is to provide organizations with a reliable and efficient way to manage their properties, tenants, and lease-related activities. By minimizing manual tasks and ensuring accurate, up-to-date information, the system supports better compliance, enhances communication, and enables teams to focus on delivering a seamless experience to tenants and stakeholders.

DEVELOPMENT PHASE

Creating Developer Account:

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

The screenshot shows the 'Sign up for your Developer Edition' page. The left side features a blue header with the Salesforce logo and a call-to-action: 'Build enterprise-quality apps fast and get hands-on with Agentforce and Data Cloud.' Below this, there's a list of benefits: 'Sign up for your Developer Edition.', followed by a bulleted list: '✓ Build apps fast with drag-and-drop tools', '✓ Go further with Apex code', '✓ Build AI agents with Agentforce', '✓ Harmonize your data with Data Cloud', '✓ Ground Agentforce with structured and unstructured data', and '✓ Integrate with anything using APIs'. The right side contains form fields for 'First name' (Azwin), 'Last name' (Muhammed), 'Job title' (Student), 'Work email' (azwinmuhammed.23@), 'Company' (Kumaraguru College), 'Country/Region' (India), and a checkbox for agreeing to the 'Main Services Agreement – Developer Services and Salesforce Program Agreement'. A note at the bottom states: 'Your org may be provisioned on or migrated to Hyperforce, Salesforce's public cloud infrastructure.'

Sign up for your Developer Edition

A free Salesforce Platform environment with Agentforce and Data Cloud

First name Last name

Job title Work email

Company Country/Region

Your org may be provisioned on or migrated to Hyperforce, Salesforce's public cloud infrastructure.

I agree to the Main Services Agreement – Developer Services and Salesforce Program Agreement. I acknowledge, as described in the Developer Documentation: (1) the Developer Edition includes autonomous and other generative AI features; and (2) Salesforce may limit use of those features and the org, and may terminate any org that has been inactive for 45 days.

- Created objects: Property, Tenant, Lease, Payment

The screenshot shows the Salesforce Setup interface with the 'Object Manager' selected. The left sidebar lists various configuration options for the 'property' object, including 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 panel displays the 'Details' tab for the 'property' object, which includes fields for Description, API Name (property__c), Singular Label (property), Plural Label (property), Enable Reports (checked), Track Activities (checked), Track Field History (checked), Deployment Status (Deployed), Help Settings (Standard salesforce.com Help Window), and a 'Edit' and 'Delete' button.

The screenshot shows the Salesforce Setup interface with the 'Object Manager' selected. The left sidebar lists various configuration options for the 'Tenant' object, including 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 panel displays the 'Details' tab for the 'Tenant' object, which includes fields for Description, API Name (Tenant__c), Singular Label (Tenant), Plural Label (Tenants), Enable Reports (checked), Track Activities (checked), Track Field History (checked), Deployment Status (Deployed), Help Settings (Standard salesforce.com Help Window), and a 'Edit' and 'Delete' button.

The screenshot shows the Salesforce Setup interface with the 'Object Manager' selected. The top navigation bar includes 'Setup', 'Home', 'Object Manager', and a search bar. The main content area displays the 'lease' object details. On the left, a sidebar lists various configuration options: 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, and Restriction Rules. The right side shows the 'Details' tab for the 'lease' object, which includes fields for API Name ('lease__c'), Singular Label ('lease'), and Plural Label ('lease'). It also shows settings for Enable Reports (checked), Track Activities (checked), Track Field History (checked), Deployment Status ('Deployed'), Help Settings, and a link to the 'Standard salesforce.com Help Window'. Action buttons 'Edit' and 'Delete' are located at the top right of the details section.

The screenshot shows the Salesforce Setup interface with the 'Object Manager' selected. The top navigation bar includes 'Setup', 'Home', 'Object Manager', and a search bar. The main content area displays the 'Payment for tenant' object details. On the left, a sidebar lists various configuration options: 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, and Restriction Rules. The right side shows the 'Details' tab for the 'Payment for tenant' object, which includes fields for API Name ('Payment_for_tenant__c'), Singular Label ('Payment for tenant'), and Plural Label ('Payment'). It also shows settings for Enable Reports (checked), Track Activities (checked), Track Field History (checked), Deployment Status ('Deployed'), Help Settings, and a link to the 'Standard salesforce.com Help Window'. Action buttons 'Edit' and 'Delete' are located at the top right of the details section.

- Configured fields and relationships

SETUP > OBJECT MANAGER
property

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	CreatedBy	Lookup(User)		
Last Modified By	LastModifiedBy	Lookup(User)		
Name	Name__c	Text(25)		
Owner	OwnerId	Lookup(User,Group)	✓	
property	property__c	Lookup(property)	✓	
property Name	Name	Text(90)	✓	
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)		
check for payment	check_for_payment__c	Picklist		
Created By	CreatedBy	Lookup(User)		
Last Modified By	LastModifiedBy	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)	✓	
Payment date	Payment_date__c	Date		
Payment Name	Name	Text(90)	✓	

Setup > Object Manager

lease

Fields & Relationships		7 Items, Sorted by Field Label				
		FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Details	Fields & Relationships	Created By	CreatedById	Lookup(User)		
	Page Layouts	End date	End_date_c	Date		
	Lightning Record Pages	Last Modified By	LastModifiedById	Lookup(User)		
	Buttons, Links, and Actions	lease Name	Name	Text(80)		
	Compact Layouts	Owner	OwnerId	Lookup(User/Group)		
	Field Sets	property	property_c	Lookup(property)		
	Object Limits	start date	start_date_c	Date		
	Record Types					
	Related Lookup Filters					
	Search Layouts					
	List View Button Layout					
	Restriction Rules					
	Scoping Rules					

Setup > Object Manager

Tenant

Fields & Relationships		7 Items, Sorted by Field Label				
		FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Details	Fields & Relationships	Created By	CreatedById	Lookup(User)		
	Page Layouts	Email	Email_c	Email		
	Lightning Record Pages	Last Modified By	LastModifiedById	Lookup(User)		
	Buttons, Links, and Actions	Owner	OwnerId	Lookup(User/Group)		
	Compact Layouts	Phone	Phone_c	Phone		
	Field Sets	status	status_c	Picklist		
	Object Limits	Tenant Name	Name	Text(80)		
	Record Types					
	Related Lookup Filters					
	Search Layouts					
	List View Button Layout					
	Restriction Rules					
	Scoping Rules					

- Developed Lightning App with relevant tabs

The screenshot shows the 'App Details & Branding' tab in the Lightning App Builder. On the left, a sidebar lists 'App Settings' with 'App Details & Branding' selected. The main area contains fields for 'App Name' (Lease Management), 'Developer Name' (Lease_Management), and a 'Description' box (Application to efficiently handle the processes related to leasing real estate properties). It also includes an 'Image' section with a preview thumbnail and a color picker set to #0070D2, and an 'Org Theme Options' checkbox. Below this is an 'App Launcher Preview' showing a card with the app's name and description.

The screenshot shows the 'Navigation Items' tab in the Lightning App Builder. The sidebar has 'Navigation Items' selected. The main area displays two lists: 'Available Items' on the left and 'Selected Items' on the right. The 'Available Items' list includes various Salesforce objects like Accounts, Activation Targets, Activations, All Sites, Alternative Payment Methods, Analytics, App Launcher, Appointment Categories, Appointment Invitations, Approval Requests, and more. The 'Selected Items' list contains four items: Payment, Tenants, property, and lease. Navigation arrows between the lists allow users to move items between them.

The screenshot shows the 'User Profiles' section of the Lightning App Builder. On the left, a sidebar lists navigation items: App Details & Branding, App Options, Utility Items (Desktop Only), Navigation Items, and User Profiles (which is selected and highlighted in blue). The main content area has two sections: 'Available Profiles' on the left and 'Selected Profiles' on the right. The 'Available Profiles' section contains a list of user profiles with a search bar at the top. The 'Selected Profiles' section contains one profile, 'System Administrator'. Navigation arrows between the two sections allow for selection.

App Settings

App Details & Branding

App Options

Utility Items (Desktop Only)

Navigation Items

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 a web-based lease management application interface. At the top, there's a header bar with a user profile icon, a search bar containing "Search...", and various system icons including a star, plus, minus, and a bell with a red notification count of 1. Below the header, a navigation bar features a grid icon followed by the text "Lease Management". The main menu items are "Payment" (with a dropdown arrow), "Tenants" (with a dropdown arrow), "property" (with a dropdown arrow), and "lease" (with a dropdown arrow). A blue ribbon-like banner runs across the middle of the page.

Payment

Recently Viewed ▾

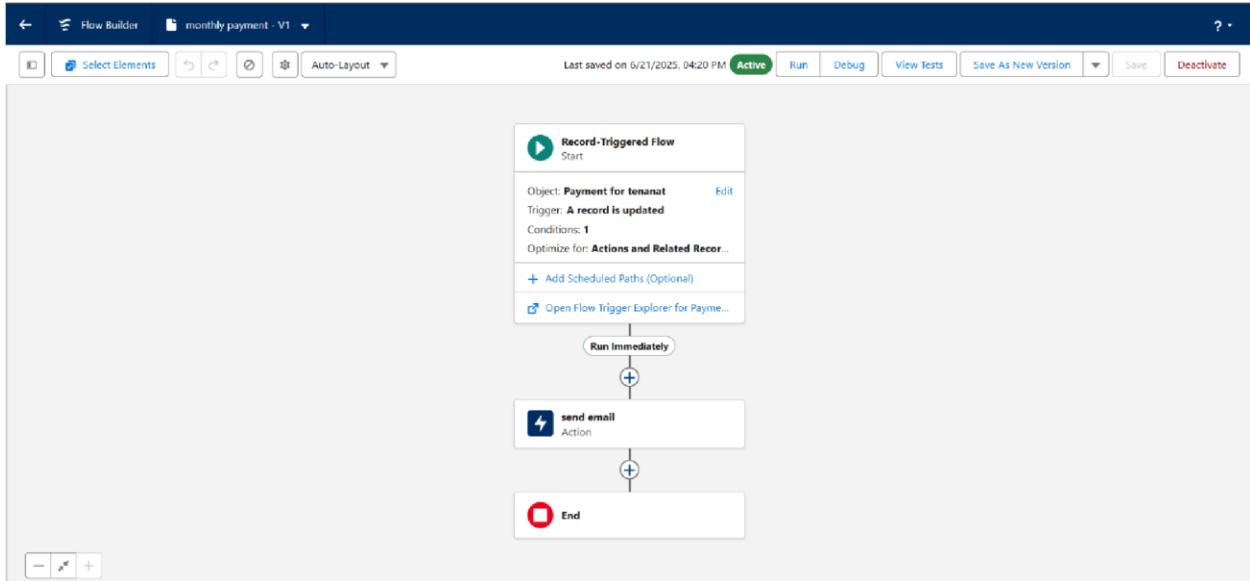
5 items • Updated a few seconds ago

	<input type="checkbox"/> Payment Name	
1	<input type="checkbox"/> Rahul	
2	<input type="checkbox"/> Jack	
3	<input type="checkbox"/> Raj	
4	<input type="checkbox"/> Sam	
5	<input type="checkbox"/> Lahari	

New Import Change Owner Assign Label

Search this list...

- Implemented Flows for monthly rent and payment success



- To create a validation rule to a Lease Object

The screenshot shows the Validation Rule Edit screen for the Lease object in the Object Manager.

Validation Rule Edit

Details: Validation Rule Name: lease_end_date, Active: checked.

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.

Quick Tips: Operators & Functions

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

Page Layout: lease Validation Rule

Validation Rule Detail:

- 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:** Sowmya Team, 6/19/2025, 5:37 AM
- Active:** ✓
- Error Location:** start date
- Modified By:** Sowmya Team, 6/26/2025, 7:47 AM

Left sidebar: Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Restriction Rules, Scoping Rules.

- Added Apex trigger to restrict multiple tenants per property

The screenshot shows the Salesforce Lease Management interface with the following details:

Page Layout: New Tenant

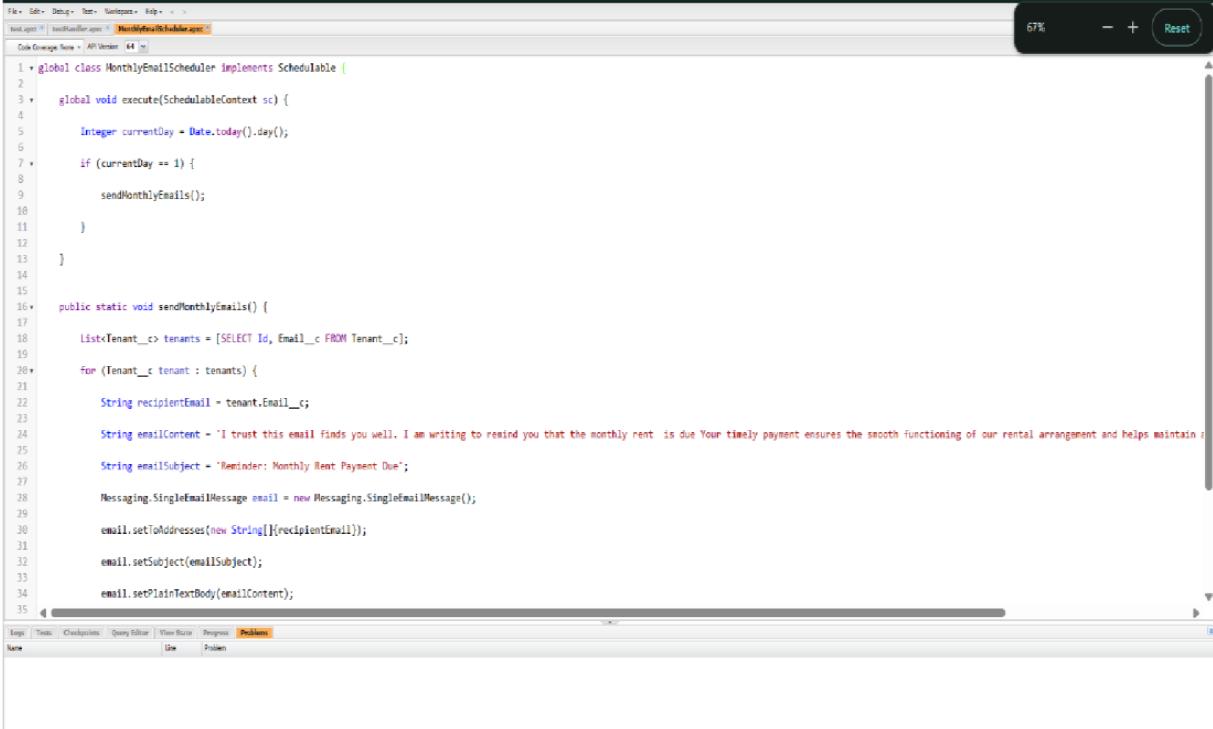
Information:

- Tenant Name:** chinu
- Email:** chinu@gmail.com
- Phone:** (empty)
- Status:** Stay
- property:** Parkside

Validation Error: We hit a snag. A tenant can have only one property.

Buttons: Cancel, Save & New, Save

- Scheduled monthly reminder emails using Apex class

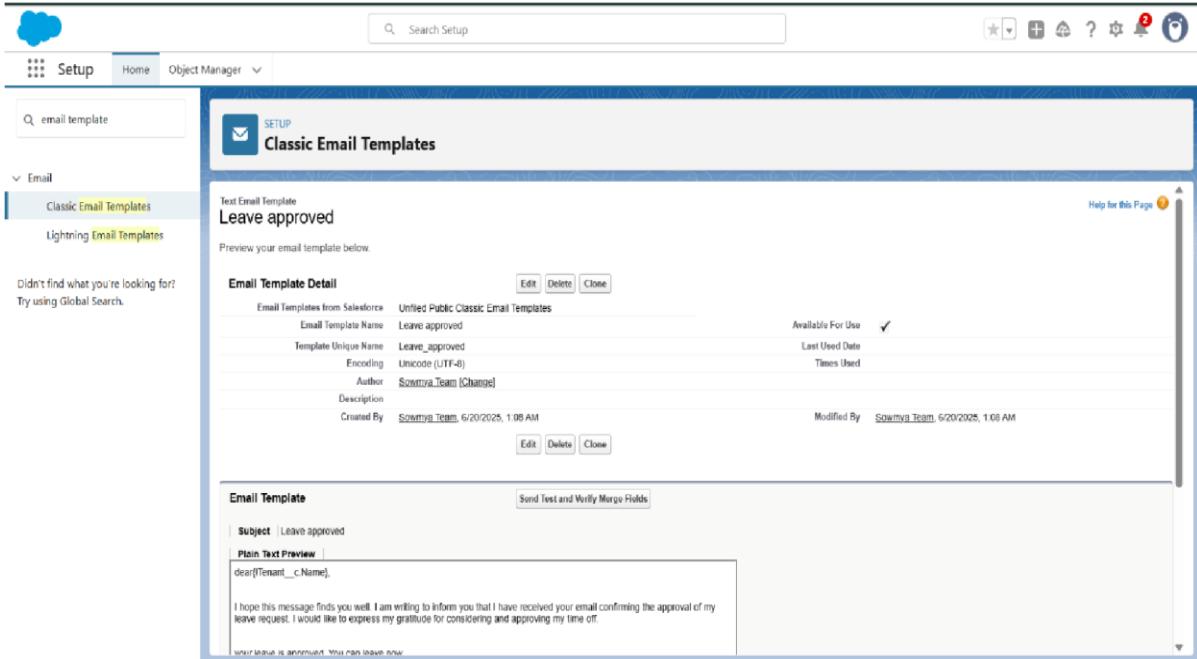


```

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

The screenshot shows the Salesforce Developer Console with the code for the `MonthlyEmailScheduler.apex` class. The code implements the `Schedulable` interface and contains a scheduled method `execute` that checks if it's the first day of the month. If so, it calls the static method `sendMonthlyEmails`. This method retrieves all `Tenant__c` records and sends a monthly rent reminder email to each tenant. The email content is a placeholder message, and the subject is "Reminder: Monthly Rent Payment Due". The code is annotated with comments explaining its purpose.

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



The screenshot shows the Salesforce Setup page under the `Email` section, specifically the `Classic Email Templates` section. A new email template named `Leave approved` is being created. The template details are as follows:

Field	Value
Email Template Name	Leave approved
Template Unique Name	Leave_approved
Encoding	Unicode (UTF-8)
Author	Sowmya Team [Change]
Description	
Created By	Sowmya Teja
Modified By	Sowmya Teja
Available For Use	<input checked="" type="checkbox"/>
Last Used Date	
Times Used	

The `Plain Text Preview` section contains the following text:

```

Subject: Leave approved
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.  

Even though it is approved, you can review now.

```

The screenshot shows the Salesforce Setup interface with the 'Email' section selected. A search bar at the top right contains 'Search Setup'. Below it, a navigation bar includes 'Setup', 'Home', and 'Object Manager'. A sidebar on the left has a search bar for 'email template' and lists 'Classic Email Templates' and 'Lightning Email Templates'. A message says ' Didn't find what you're looking for? Try using Global Search.' The main content area is titled 'Classic Email Templates' and shows a 'Text Email Template' named 'tenant leaving'. The 'Email Template Detail' section includes fields for 'Email Template Name' (tenant leaving), 'Template Unique Name' (tenant_leaving), 'Encoding' (Unicode (UTF-8)), 'Author' (Sowmya Team [Change]), 'Description' (None), 'Created By' (Sowmya Team), 'Modified By' (Sowmya Team), and status indicators for 'Available For Use' (checked), 'Last Used Date' (N/A), and 'Times Used' (0). Below this is a preview pane with a subject line 'request for approve the leave' and a plain text preview: 'Dear {!Tenant__c.CreatedBy}, Please approve my leave'.

This screenshot shows the same Salesforce Setup interface as the first one, but the 'Email' section is not selected. The 'Classic Email Templates' section is highlighted. The main content area shows a 'Text Email Template' named 'Leave rejected'. The 'Email Template Detail' section includes fields for 'Email Template Name' (Leave rejected), 'Template Unique Name' (Leave_rejected), 'Encoding' (Unicode (UTF-8)), 'Author' (Sowmya Team [Change]), 'Description' (None), 'Created By' (Sowmya Team), 'Modified By' (Sowmya Team), and status indicators for 'Available For Use' (checked), 'Last Used Date' (N/A), and 'Times Used' (0). Below this is a preview pane with a subject line 'Leave rejected' and a 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	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

Plain Text Preview

Subject: Urgent: Monthly Rent Payment Reminder

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. Please be sure to make arrangements to pay it in full by the end of the month.

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

Classic Email Templates

tenant payment

Email Template Detail

Email Templates from Salesforce	Unified Public Classic Email Templates
Email Template Name	tenant payment
Template 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}},

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: TenantLeaving

Process Definition Detail

Process Name	TenantApproval	Active
Unique Name	TenantApproval	Next Automated Approver Determined By
Description		
Entry Criteria	Tenant: status EQUALS Stay	
Record Editability	Administrator ONLY	Allow Submitters to Recall Approval Requests

Approval Assignment Email Template: Initial Submitters: Tenant Owner
Created By: Sowmya Team, 6/23/2025, 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 Edit	1	Step 1			User: Sowmya Team	Final Rejection

For Check for Vacant:

Approval Processes
Tenant: check_for_vacant

Process Definition Detail

Process Name	check_for_vacant	Active
Unique Name	check_for_vacant	Next Automated Approver Determined By
Description		
Entry Criteria	Tenant: status EQUALS Leaving	
Record Editability	Administrator ONLY	Allow Submitters to Recall Approval Requests

Approval Assignment Email Template: Leave approved
Initial Submitters: Tenant Owner
Created By: Sowmya Team, 6/20/2025, 3:18 AM Modified By: Sowmya Team, 6/26/2025, 11:02 PM

Initial Submission Actions

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

Approval Steps

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

- Apex Trigger

Create an Apex Trigger

The screenshot shows the Salesforce IDE interface. In the top navigation bar, the tabs are: File, Edit, Debug, Test, Workspace, Help, Go To. Below the tabs, there are dropdown menus for Code Coverage (None), API Version (54), and a Go To field. The main editor window contains the following Apex code:

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

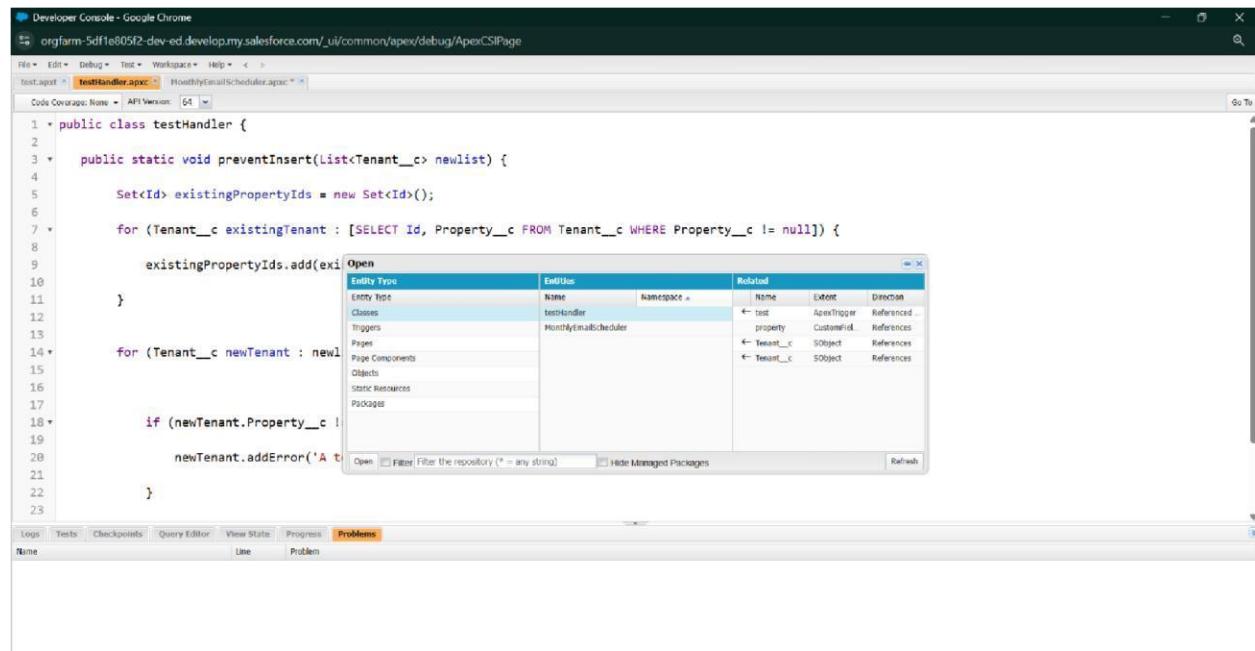
A modal dialog titled "Open" is displayed over the code editor. The "Entity Type" dropdown is set to "Triggers". The table lists one entry: "test" under the "Name" column and "Namespace" column. There are also columns for "Related", "Name", "Extent", and "Direction". At the bottom of the modal are buttons for "Open", "Filter", "Hide Managed Packages", and "Refresh".

The screenshot shows the Developer Console in Google Chrome. The URL is https://orgfarm-5df1e805f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The browser menu bar includes File, Edit, Debug, Test, Workspace, Help, and a magnifying glass icon. The developer console tabs at the top are: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is selected. The main area displays the same Apex code as the previous screenshot:

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

Below the code, there is a table with columns: Name, Line, and Problem. The table is currently empty.

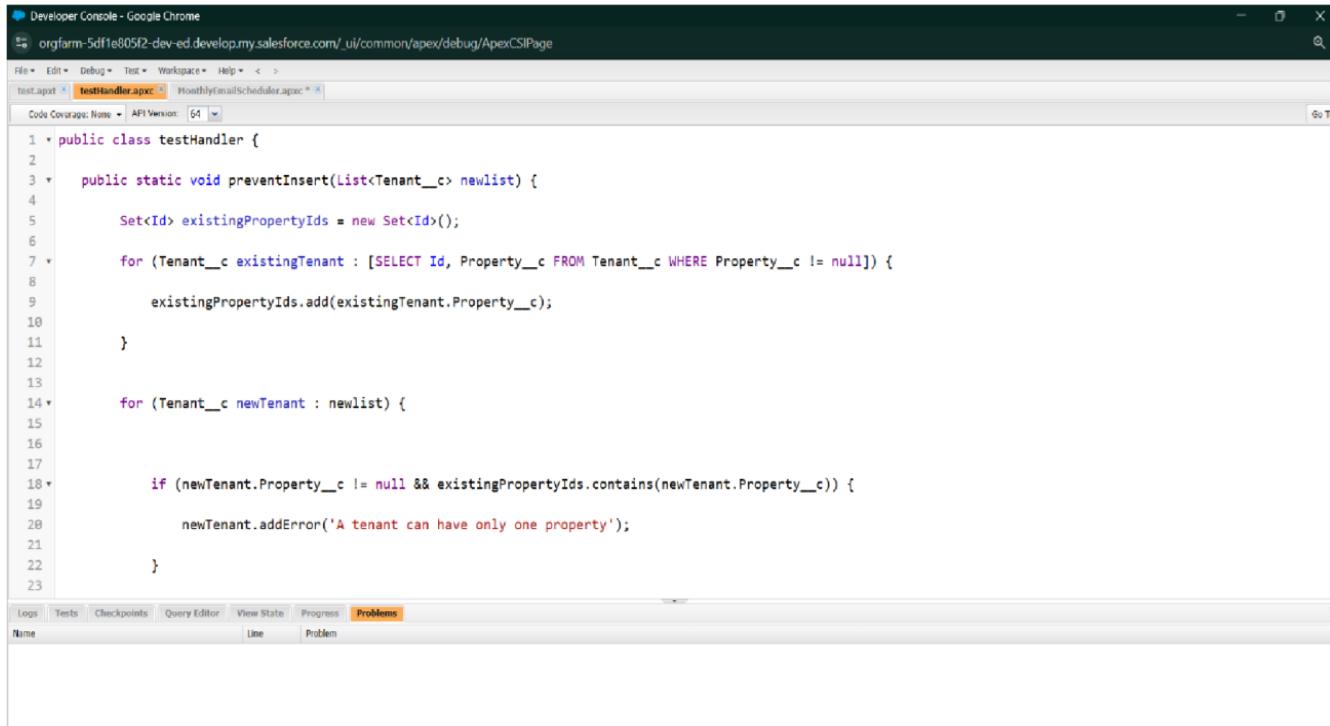
Create an Apex Handler class



The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is `orgfarm-5df1e805f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage`. The tab bar has `test.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             existingPropertyIds.add(existingTenant.Property__c);  
9         }  
10        for (Tenant__c newTenant : newList) {  
11            if (newTenant.Property__c != null && existingPropertyIds.contains(newTenant.Property__c)) {  
12                newTenantaddError('A tenant can have only one property');  
13            }  
14        }  
15    }  
16}  
17  
18    if (newTenant.Property__c != null && existingPropertyIds.contains(newTenant.Property__c)) {  
19        newTenantaddError('A tenant can have only one property');  
20    }  
21}  
22}
```

A dependency graph window is open over the code editor, showing relationships between classes and triggers. The graph includes entities like `testHandler`, `MonthlyEmailScheduler`, `ApexTrigger`, `CustomField`, `Object`, `Subject`, and `Tenant__c`.



The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is `orgfarm-5df1e805f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage`. The tab bar has `test.apc` selected. The code editor contains the same Apex class as the previous screenshot, but with a bug fix added at line 18:

```
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             existingPropertyIds.add(existingTenant.Property__c);  
9         }  
10        for (Tenant__c newTenant : newList) {  
11            if (newTenant.Property__c != null && existingPropertyIds.contains(newTenant.Property__c)) {  
12                newTenantaddError('A tenant can have only one property');  
13            }  
14        }  
15    }  
16}  
17  
18    if (newTenant.Property__c != null && existingPropertyIds.contains(newTenant.Property__c)) {  
19        newTenantaddError('A tenant can have only one property');  
20    }  
21}  
22}
```

● FLOWS

Flow Builder monthly payment - V1

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

Record-Triggered Flow Start

Object: Payment for tenantat Trigger: A record is updated Conditions: 1 Optimize for: Actions and Related Recor... + Add Scheduled Paths (Optional) Open Flow Trigger Explorer for Payne...

Run Immediately

send email Action

End

Configure Start

Field: check for payment Operator: Equals Value: A_a Paid

+ Add Condition

When to Run the Flow for Updated Records

- Every time a record is updated and meets the condition requirements
- Only when a record is updated to meet the condition requirements

Optimize Flow

Optimize the Flow for:

Fast Field Updates Update fields on the record that triggers the flow to run. This high-performance flow runs **before the record is saved** to the database.

Actions and Related Records Update any record and perform actions, like send an email. This more flexible flow runs **after the record is saved** to the database.

Is this flow making an external callout or connecting to an external system? An asynchronous path is required for flows that involve external systems.

Add Asynchronous Path

Flow Builder monthly payment - V1

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

Record-Triggered Flow Start

Object: Payment for tenantat Trigger: A record is updated Conditions: 1 Optimize for: Actions and Related Recor... + Add Scheduled Paths (Optional) Open Flow Trigger Explorer for Payne...

Run Immediately

send email Action

End

Configure Start

Select Object

* Object: Payment for tenantat

Configure Trigger

Trigger the Flow When:

- A record is created
- A record is updated
- A record is created or updated
- A record is deleted

Set Entry Conditions

Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.

If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the **Only when a record is updated to meet the condition requirements** option for When to Run the Flow for Updated Records.

Condition Requirements

All Conditions Are Met (AND)

Field	Operator	Value

- Schedule class:
Create an Apex Class

Developer Console - Google Chrome

File Edit Debug Test Workspace Help < >

test.apex testHandler.apxc MonthlyEmailScheduler.apxc

Code Coverage: None API Version: 64 Go To

```
1 * global class MonthlyEmailScheduler implements Schedulable {  
2  
3     global void execute(SchedulableContext sc) {  
4  
5         Integer currentDay = Date.today().day();  
6  
7         if (currentDay == 1) {  
8  
9             sendMonthlyEmails();  
10        }  
11    }  
12  
13 }  
14  
15  
16 public static void sendMonthlyEmail  
17 {  
18     List<Tenant__c> tenants = [SEL  
19  
20         for (Tenant__c tenant : tenants)  
21  
22             String recipientEmail = tenant.Email__c;  
23  
24 }
```

Open Entity Type Entity Related

Entity Type	Name	Namespace	Name	Extent	Direction
Entity Type	testHandler		Email	CronTrigger	Referenced...
Classes			← Tenant__c	SObject	References
Triggers			← Tenant__c	SObject	References
Pages					
Page Components					
Objects					
Static Resources					
Packages					

Open Filter Hide Managed Packages Refresh

Logs Tests Checkpoints Query Editor View Status Progress Problems

The screenshot shows the Google Chrome Developer Console with the URL <https://orgfarm-5df11e805f2-dev-ed.develop.my.salesforce.com/.ui/common/apex/debug/ApexCSPage>. The console displays an Apex class named `MonthlyEmailScheduler` with a static method `sendMonthlyEmails`. The code implements a scheduled apex job that sends monthly reminder emails to tenants.

```
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});
        }
    }
}
```

Schedule Apex class

The screenshot shows the Salesforce Setup interface under the Apex Classes section. The class **MonthlyEmailScheduler** is selected. The code editor displays 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
9     public static void sendMonthlyEmails() {
10        List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
11        for (Tenant__c tenant : tenants) {
12            ...
13        }
14    }
15
16    public static void sendMonthlyEmails() {
17        List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
18        for (Tenant__c tenant : tenants) {
19            ...
20        }
21    }
22}
```

The screenshot shows the Lease Management application's Tenant record for **Aswini**. The Details tab is active. The contact information is as follows:

- Tenant Name: Aswini
- Email: aswiniamaraadi15@gmail.com
- Phone: (905) 223-5567
- Status: Leaving
- Property: Imran

The Activity sidebar is open, showing the following options:

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

The activity feed is currently empty.

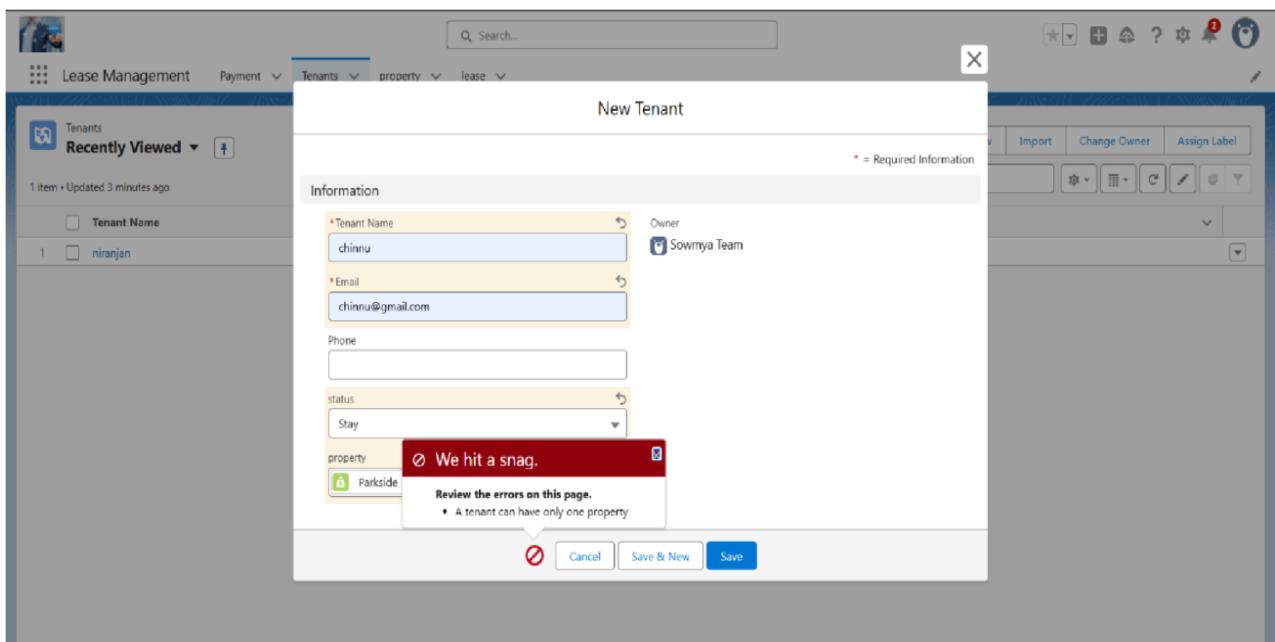
The screenshot shows a CRM interface for 'Lease Management'. A green notification bar at the top right says 'Tenant was submitted for approval.' Below it, the main area displays a 'Details' tab for a tenant named 'Aswini'. The form includes fields for Tenant Name, Email, Phone, Status (Leaving), and Property (Imran). The 'Owner' field is set to 'Sowmya Team'. At the bottom, there are 'Cancel' and 'Save' buttons. To the right, an 'Activity' sidebar shows no upcoming or overdue activities. The system status bar at the bottom indicates it's 12:46 on June 27, 2025.

This screenshot shows a process instance step titled 'Tenant Approval [Approved]'. It displays details about the approval: Submitter (Sowmya Team), Date Submitted (Jun 27, 2025), Actual Approver (Sowmya Team), and Assigned To (Sowmya Team). Below this, an 'Approval Details' section shows the tenant name (Aswini), property (Imran), and owner (Sowmya Team). On the right, a 'Notifications' sidebar lists several recent notifications related to tenant approvals, including ones from Aswini and Kiran.

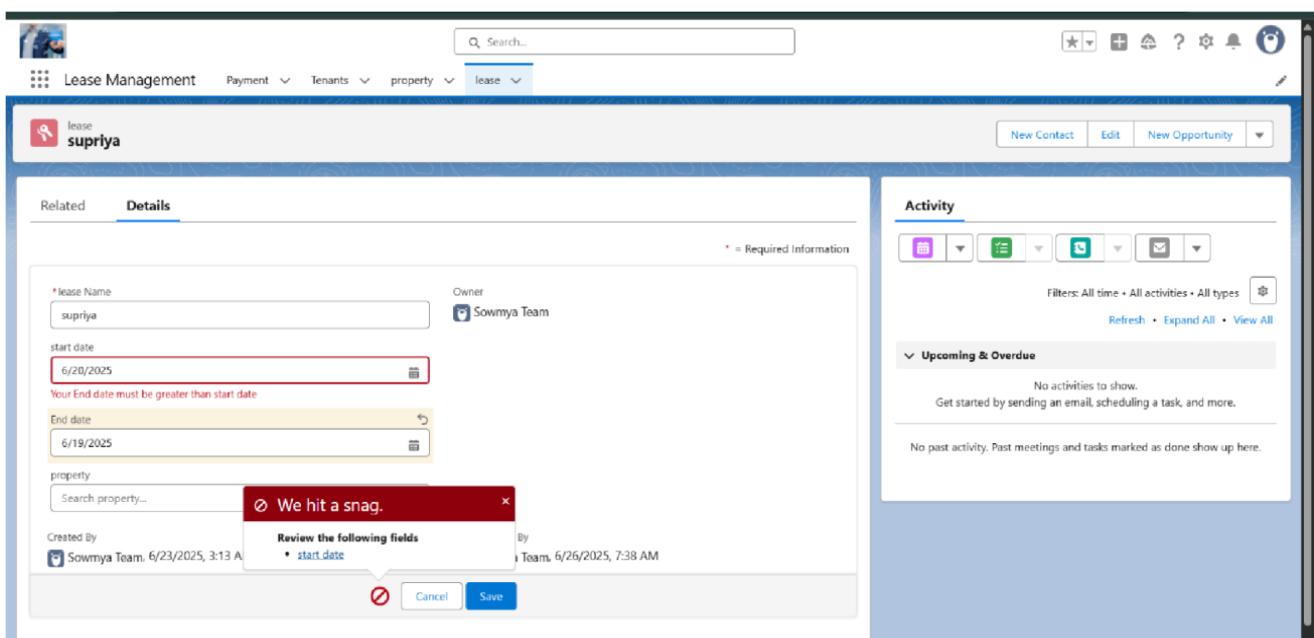
FUNCTIONAL AND PERFORMANCE TESTING

Performance Testing

- Trigger validation by entering duplicate tenant-property records



- Validation Rule checking



Lease Management

lease supriya

Related Details

* Required Information

lease Name: supriya

Owner: Sowmya Team

start date: 6/20/2025

Your End date must be greater than start date.

end date: 6/19/2025

property: Search property...

Created By: Sowmya Team, 6/23/2025, 3:13 AM

We hit a snag.

Review the following fields:

- start_date

By Team, 6/26/2025, 7:38 AM

Activity

No activities to show.

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

- Test flows on payment update

Flow Builder

monthly payment - V1

Test: Payment Update Passed

Record-Triggered Flow Start

Run Immediately

send email Action

End

- +

Test Run Details

Assertions All Details

> Assertion 1 Passed

- 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 placeholder 'Search...'. Below it, a navigation bar includes 'Lease Management', 'Payment', 'Tenants', 'property', 'lease', and a dropdown for 'niranjan Approval'. The main area displays a 'Details' card for a tenant named 'niranjan'. The card includes fields for 'Tenant Name' (niranjan), 'Email' (niranjan1506@gmail.com), 'Phone' (empty), 'status' (Stay), and 'property' (Parkside Lofts). It also shows 'Created By' (Sowmya Team) and 'Last Modified By' (Sowmya Team) with their respective timestamps. At the bottom are 'Cancel' and 'Save' buttons. To the right, a sidebar titled 'Notifications' lists several items: 'Approval request for the tenant is approved niranjan' (a few seconds ago), 'Approval request for the tenant is rejected niranjan' (Jun 23, 2025, 4:29 PM), 'Approval request for the tenant is approved niranjan' (Jun 23, 2025, 4:25 PM), 'Approval request for the tenant is approved niranjan' (Jun 23, 2025, 4:14 PM), and a promotional message for 'New Guidance Center learning resource available'.

This screenshot shows the same CRM interface for 'Lease Management' under the 'Tenants' tab. The 'Approval History' section displays a table with six rows, each representing an approval step. The columns are 'Step Name', 'Date', 'Status', and 'Assigned To'. The steps are: Step 1 (Approved, 6/25/2025, 5:39 AM, Sowmya Team), Approval Request Submitted (Submitted, 6/25/2025, 5:39 AM, Sowmya Team), Step 1 (Rejected, 6/23/2025, 3:59 AM, Sowmya Team), Approval Request Submitted (Submitted, 6/23/2025, 3:58 AM, Sowmya Team), Step 1 (Approved, 6/23/2025, 3:55 AM, Sowmya Team), and Approval Request Submitted (Submitted, 6/23/2025, 3:55 AM, Sowmya Team). Below this is a 'View All' button. The 'Payment' section shows two entries: 'Jack' and 'Rahul', with a 'New' button to add more. A 'View All' button is also present here. On the right side, there's a sidebar with buttons for 'New Contact', 'Edit', and 'New Opportunity'. A message in the sidebar says '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.'

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	Tenants	Map	

Below this are sections for 'Web Tabs' and 'Visualforce Tabs', both currently empty.

- Email alerts

The screenshot shows the 'Lease Management' application's 'Approval History' page. It lists 8 items sorted by 'Is Pending' and updated a few seconds ago. The table columns are 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

- Leave approved



Azwin Muhammed

To: Azwin Muhammed Kk. 23BDS006



Reply

Reply all

Forward



...

Sat 13-09-2025 17:17

CAUTION: This email has originated from outside of the organization. Do not click links or open attachments unless you recognize the sender and know the content is safe.

dear,

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

Reply

Forward

- Leave rejected



Azwin Muhammed

To: Azwin Muhammed Kk. 23BDS006



Reply

Reply all

Forward



...

Sat 13-09-2025 17:19

CAUTION: This email has originated from outside of the organization. Do not click links or open attachments unless you recognize the sender and know the content is safe.

Dear ,

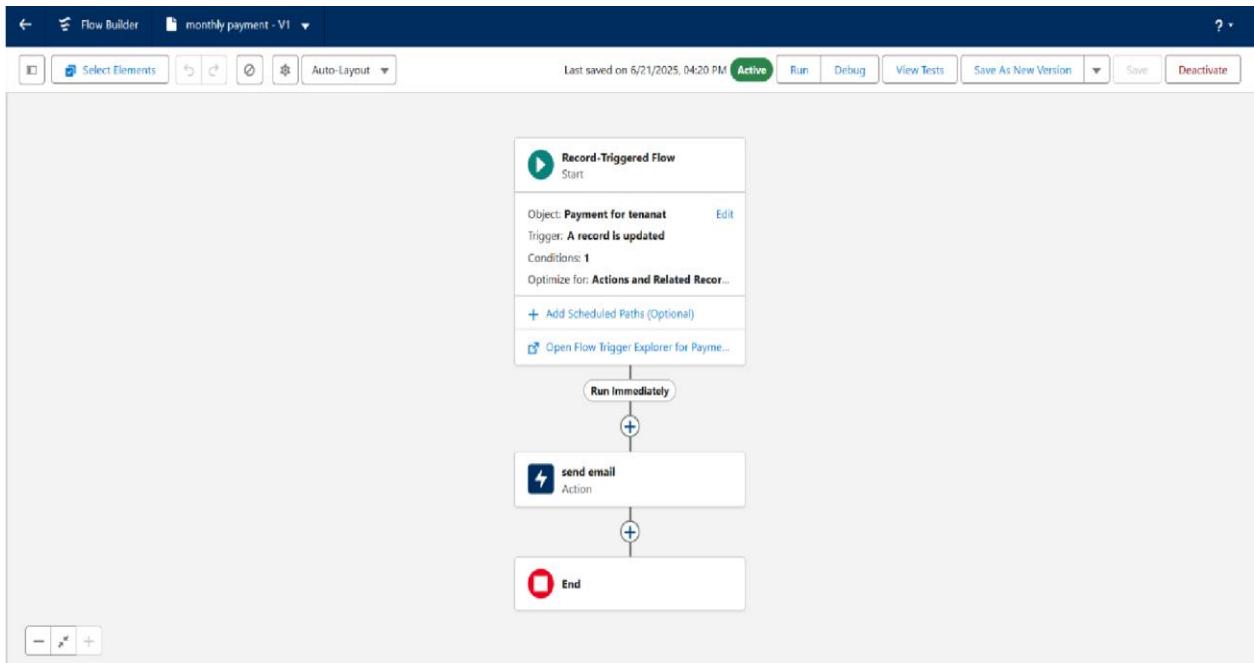
I hope this email finds you well. Your contract has not ended. So we can't approve your leave

your leave has rejected

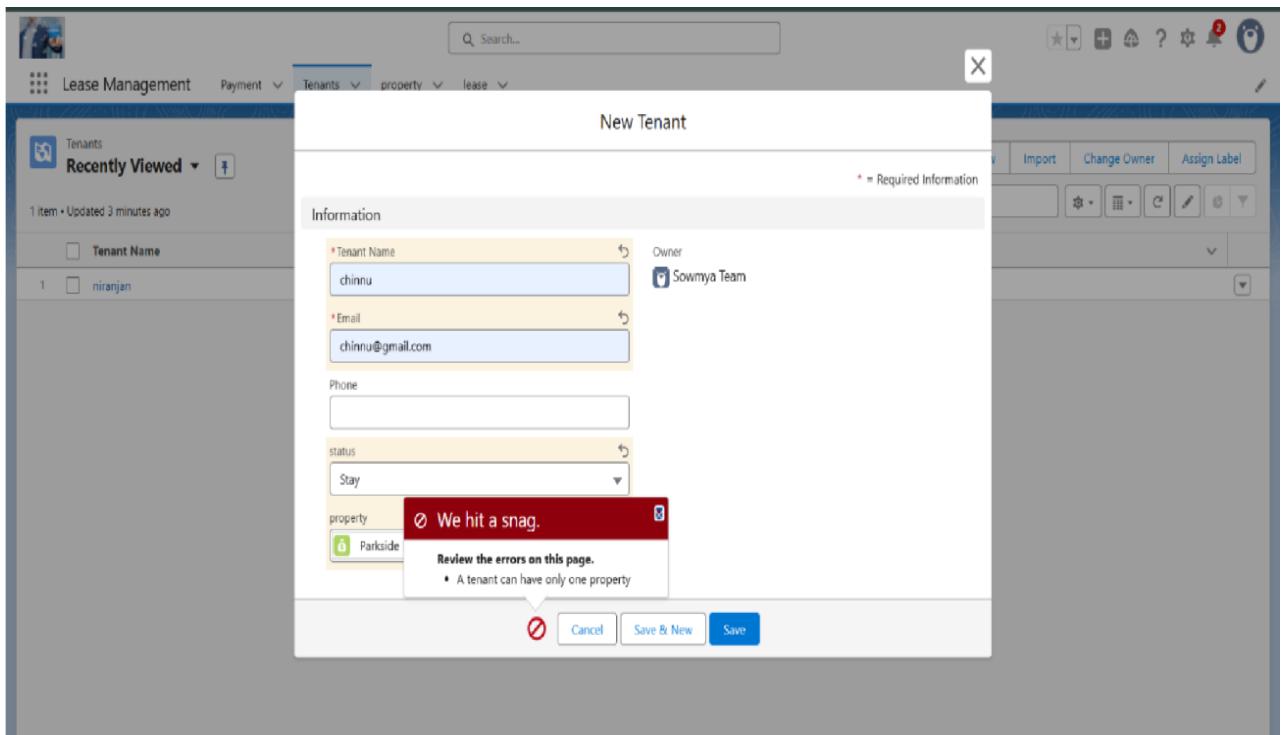
Reply

Forward

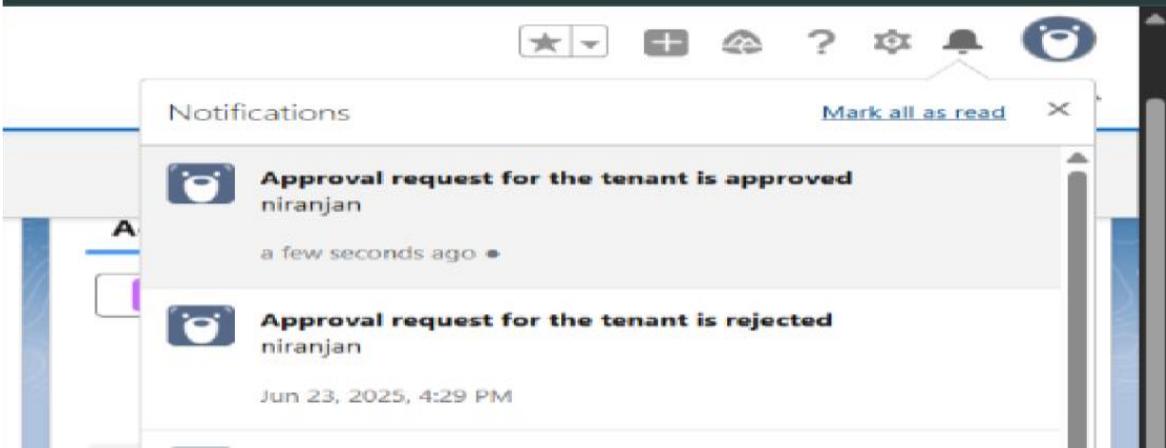
- Flow runs



- Trigger error messages



- Approval process notifications



ADVANTAGES & DISADVANTAGES

ADVANTAGES

- **Centralized Management** – Combines property, tenant, lease, and payment details in one platform, reducing duplication and errors.
- **Process Automation** – Automates approvals, reminders, and notifications, saving time and reducing manual effort.
- **Better Decision-Making** – Provides real-time reports and dashboards to track leases, payments, and occupancy status effectively.

DISADVANTAGES

- **High Initial Costs** – Requires investment in Salesforce licensing, customization, and user training.
 - **Learning Curve** – Users may need time and training to become familiar with the system's features.
 - **Internet Dependency** – Being cloud-based, it requires reliable internet connectivity for uninterrupted access.
-

CONCLUSION

The Lease Management System effectively streamlines leasing operations through a structured and automated Salesforce application. It enhances efficiency, ensures accurate data management, and improves communication between administrators 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});  
    }  
}  
}
```