

LEASE MANAGEMENT

COLLEGENAME:KGCollege ofArtsAndScience

TEAM ID:NM2025TMID23624

TEAM SIZE:4

TEAM MEMBERS:

Team Leader Name:Narmathashree T

Email:2326kb29@kgcas.com

Team Member:Nishanth C

Email:2326kb30@kgcas.com

Team Member:Nithish S

Email:2326kb31@kgcas.com

Team Member:Nithyashree S

Email:2326kb32@kgcas.com

1. INTRODUCTION

1.1 Project Overview

TheLease Management System is aSalesforce-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.



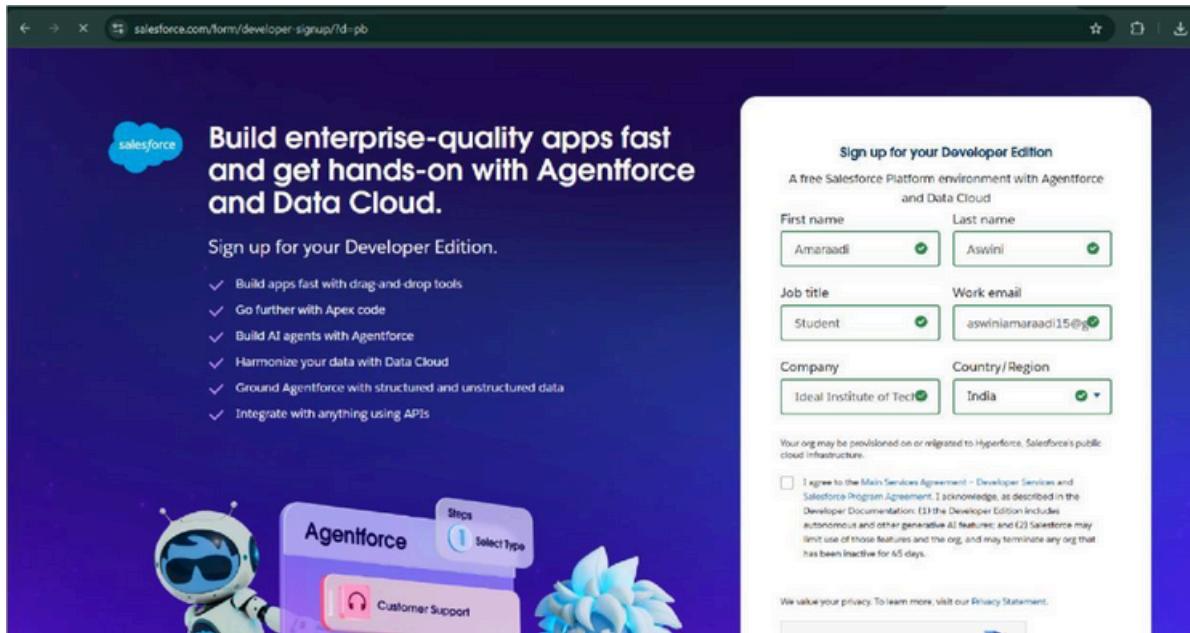
1.2 Purpose

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

- Here are 8 key points from the screenshot:
- Object Name: property
- API Name: property__c
- Custom Object: Yes (✓)
- Singular Label: property
- Plural Label: property
- Enable Reports: Checked (✓)
- Track Activities and Field History: Enabled (✓)
- Deployment Status: Deployed

orgfarm-5df1e805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK00000zTbN/Details/view

Search Setup

up Home Object Manager

OBJECT MANAGER

t

Details

Edit

Description

Relationships

outs

Record Pages

Links, and Actions

Layouts

mits

ypes

lookup Filters

lyouts

Button Layout

in Rules

Rules

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

This screenshot shows the 'Object Manager' interface in Salesforce. On the left, there's a sidebar with various tabs like 'Relationships', 'outs', 'Record Pages', etc. The main area is titled 'Details' for the 'Tenant_c' object. It shows the API name as 'Tenant_c', which is a custom object. The singular label is 'Tenant' and the plural label is 'Tenants'. Under 'Enable Reports', several tracking options are checked: 'Track Activities', 'Track Field History', and 'Deployment Status' (set to 'Deployed'). The 'Help Settings' link points to the standard salesforce.com Help Window.

orgfarm-5df1e805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK00000zTuJ/Details/view

Search Setup

up Home Object Manager

OBJECT MANAGER

Details

Edit

Description

Relationships

outs

Record Pages

Links, and Actions

Layouts

mits

ypes

lookup Filters

lyouts

Button Layout

in Rules

Rules

API Name: Lease_c

Custom

Singular Label: Lease

Plural Label: Lease

Enable Reports

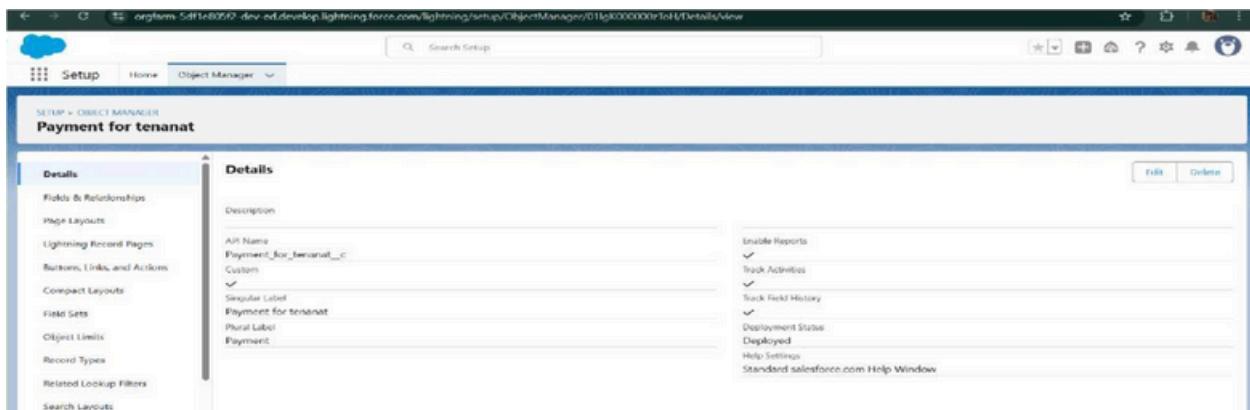
✓ Track Activities

✓ Track Field History

✓ Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

This screenshot shows the 'Object Manager' interface in Salesforce. The layout is identical to the previous one, with a sidebar and a main 'Details' section for the 'Lease_c' object. The API name is 'Lease_c', which is also a custom object. The singular label is 'Lease' and the plural label is 'Lease'. The deployment status is 'Deployed'. The help settings link again points to the standard salesforce.com Help Window.



- Configured fields and relationships

| Fields & Relationships | | | | | |
|-----------------------------|------------------|------------------|-----------------------|-------------------|---------|
| | FIELD LABEL | FIELD NAME | DATA TYPE | CONTROLLING FIELD | INDEXED |
| Relationships | Address | Address__c | Long Text Area[32768] | | |
| Record Pages | Created By | CreatedById | Lookup(User) | | |
| Buttons, Links, and Actions | Last Modified By | LastModifiedById | Lookup(User) | | |
| Layouts | Name | Name__c | Text(25) | | |
| Filters | Owner | OwnerId | Lookup(User,Group) | | ✓ |
| Lookup Filters | property | property__c | Lookup(property) | | ✓ |
| Buttons | propertyName | Name | Text(80) | | ✓ |
| Record Layout | sfqt | sfqt__c | Text(18) | | |
| Rules | Type | Type__c | Picklist | | |
| Triggers | | | | | |

SETUP > OBJECT MANAGER
lease

Fields & Relationships
7 Items, Sorted by Field Label

| FIELD LABEL | FIELD NAME | DATA TYPE | CONTROLLING FIELD | INDEXED |
|------------------|----------------|--------------------|-------------------|---------|
| Created By | CreatedBy | Lookup(User) | | |
| End date | End_date_c | Date | | |
| Last Modified By | LastModifiedBy | 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 | CreatedBy | Lookup(User) | | |
| Email | Email_c | Email | | |
| Last Modified By | LastModifiedBy | 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

The screenshot shows the 'App Details & Branding' section of the Lightning App Builder. On the left, a sidebar lists 'App Options', 'Utility Items (Desktop Only)', 'Navigation Items', and 'User Profiles'. The main area has tabs for 'App Details & Branding' (selected), 'App Details', 'App Branding', and 'Org Theme Options'. Under 'App Details', fields include 'App Name' (Lease Management), 'Developer Name' (Lease_Management), and 'Description' (Application to efficiently handle the processes related to leasing real estate properties.). Under 'App Branding', there is an 'Image' (Thumbnail of a person in a suit), a 'Primary Color Hex Value' (#0070D2), and a checkbox for 'Use the app's image and color instead of the org's custom theme'. A preview of the app launcher is shown on the right.

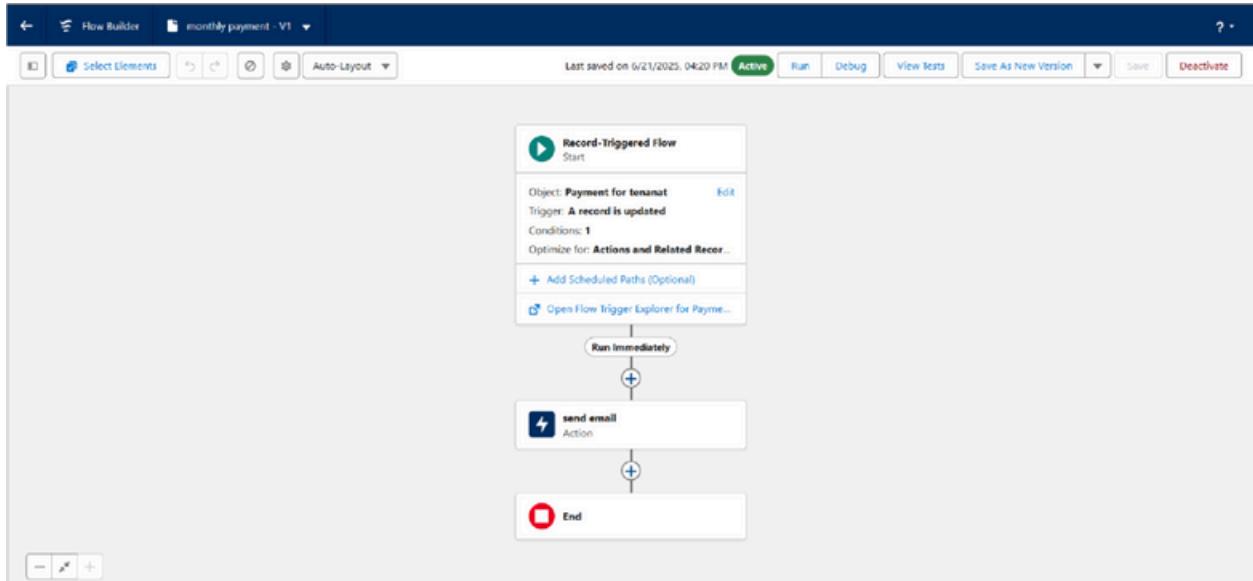
- App Name is “Lease Management”.
- Developer Name is “Lease_Management”.
- Description explains handling leasing real estate processes.
- Primary Color Hex Value is “#0070D2”.
- App Launcher Preview displays app name, image, and description.

The screenshot shows the 'Navigation Items' section of the Lightning App Builder. The sidebar includes 'App Details & Branding', 'App Options', 'Utility Items (Desktop Only)' (selected), and 'User Profiles'. The main area shows a list of 'Available Items' on the left and 'Selected Items' on the right. The 'Available Items' list contains items 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 'Payment', 'Tenants', 'property', and 'lease', with up and down arrows for reordering.

The screenshot shows the Lightning App Builder interface with the URL <https://orgfarm-5d1fe005f2-dev-ed.lightning.force.com/visualEditor/appBuilder.app?id=00ug000003NbelIQAS&relUrl=https%3A%2F%2Forgfarm-5d1fe005f2-dev-ed.lightning...>. The page title is "Lightning App Builder". The left sidebar has sections: App Details & Branding, App Options, Utility Items (Desktop Only), Navigation Items, and User Profiles (which is selected). The main content area is titled "User Profiles" and says "Choose the user profiles that can access this app." It shows two panels: "Available Profiles" (listing various profile types like Analytics Cloud Integration User, Analytics Cloud Security User, etc.) and "Selected Profiles" (containing "System Administrator").

The screenshot shows the Lease Management application with the URL https://orgfarm-5d1fe005f2-dev-ed.lightning.force.com/lightning/o/Payment_for_tenant__c/list?filterName=_Recent. The top navigation bar includes "Lease Management", "Payment", "Tenants", "property", and "lease". The main content area displays a "Recently Viewed" list of payments. The list header includes "Payment Name" and checkboxes. The items listed are:

| | Payment Name | Actions |
|---|--------------|------------|
| 1 | Rahul | [checkbox] |
| 2 | Jack | [checkbox] |
| 3 | Raj | [checkbox] |
| 4 | Sam | [checkbox] |
| 5 | Lahari | [checkbox] |



- Implemented Flows for monthly rent and payment success

Validation Rule Edit

Role Name: lease_end_date

Active:

Description:

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_d <= Start_date_d`

Functions:

- ABS
- ACOS
- ADDMONTHS
- AND
- ASCII
- ASIN

Quick Tips: • Operators & Functions

- To create a validation rule to a Lease Object

The screenshot shows the Salesforce Object Manager interface for the 'lease' object. On the left, a sidebar lists various setup options like Details, Fields & Relationships, Page Layouts, and Lightning Record Pages. The main content area displays a 'Validation Rule Detail' for a rule named 'lease_end_date'. The rule's formula is 'End_date_c <= start_date_c', with an error message stating 'Your End date must be greater than start date'. The rule is active and was created by the 'Sowmya Team' on June 19, 2025, at 5:37 AM.

- Added Apex trigger to restrict multiple tenants per property

The screenshot shows the 'Lease Management' application's 'Tenants' screen. A modal window titled 'New Tenant' is open, showing fields for 'Tenant Name' (set to 'chinu') and 'Email' (set to 'chinu@gmail.com'). An error message at the bottom of the form states: 'We hit a snag. Review the errors on this page. * A tenant can have only one property.' This indicates that the system is preventing the addition of a new tenant to an existing property.

- Scheduled monthly reminder emails using Apex class

The screenshot shows the Salesforce IDE interface with the code editor open. The file is named 'MonthlyEmailScheduler.apex'. The code implements the 'Schedulable' interface and contains logic to send monthly reminder emails to tenants if it's the 1st of the month. It uses the 'Messaging.SingleEmailMessage' class to prepare and send the email.

```
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 *     }
36 *
37 * }
38 *
```

- Built and tested email templates for leave request, approval, rejection, payment, and reminders
- Here are 5 key points (in 2 lines each) from the code in the screenshot:
- Apex class `MonthlyEmailScheduler` implements `Schedulable`.
- It runs scheduled jobs to send monthly reminder emails.
- `execute` method checks if the day is the 1st.
- If true, it calls the `sendMonthlyEmails()` method.
- `sendMonthlyEmails()` queries `Tenant` records.
- It fetches `Id` and `Email__c` fields from `Tenant__c` object.
- For each tenant, an email is prepared.
- Email includes subject "Reminder: Monthly Rent Payment Due".
- Uses `Messaging.SingleEmailMessage` to send mail.
- Sets recipient, subject, and email content dynamically.

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

Classic Email Templates

tenant leaving

Email Template Detail

| Email Templates from Salesforce | Unfiled Public Classic Email Templates |
|---------------------------------|--|
| 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 |

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

tenant leaving

Email Template Detail

| Email Templates from Salesforce | Unfiled Public Classic Email Templates |
|---------------------------------|--|
| 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 |

Email Template

Subject: request for approve the leave

Plain Text Preview:

```
Dear {[Tenant__c.CreatedBy}].  
Please approve my leave
```

Cloud icon

Setup Home Object Manager

Search bar: Search Setup

Classic Email Templates

Text Email Template Tenant Email

Preview your email template below.

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 | Scwmya_Team [Change] |
| Description | |
| Created By | Scwmya_Team 6/20/2025, 1:12 AM |
| Modified By | Scwmya_Team 6/20/2025, 1:12 AM |

Available For Use ✓

Last Used Date

Times Used

Email Template

Send Test and Verify Merge Fields

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.

Help for this Page

Cloud icon

Setup

Classic Email Templates

Text Email Template tenant payment

Preview your email template below.

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 | Scwmya_Team [Change] |
| Description | |
| Created By | Scwmya_Team 6/20/2025, 1:13 AM |
| Modified By | Scwmya_Team 6/20/2025, 1:13 AM |

Available For Use ✓

Last Used Date

Times Used

Email Template

Send Test and Verify Merge Fields

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.

The screenshot shows the Salesforce Setup interface with the search bar set to 'email template'. Under the 'Email' section, 'Classic Email Templates' is selected. A specific template, 'tenant payment', is highlighted. The 'Email Template Detail' section shows the following information:

- Email Template Name: tenant payment
- Template Unique Name: tenant_payment
- Encoding: Unicode (UTF-8)
- Author: Sreeniva_Team (Changed)
- Description: Untitled Public Classic Email Templates
- Created By: Sreeniva_Team, 6/20/2025, 1:13 AM
- Modified By: Sreeniva_Team, 6/20/2025, 1:13 AM
- Available For Use: checked
- Last Used Date: N/A
- Times Used: N/A

The 'Email Template' section below shows the preview of the email:

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 search bar set to 'approval'. Under the 'Data' section, 'Approval Processes' is selected. A specific process, 'Tenant: TenantApproval', is highlighted. The 'Process Definition Detail' section shows the following information:

- Process Name: TenantApproval
- Unique Name: TenantApproval
- Description: N/A
- Entry Criteria: IsStatus: status equals Stay
- Record Editability: Administrator ONLY
- Next Automated Approver Determined by: N/A
- Allow Submitters to Recall Approval Requests: unchecked
- Created By: Sreeniva_Team, 6/20/2025, 3:41 AM
- Modified By: Sreeniva_Team, 6/20/2025, 11:57 PM
- Active: checked

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 | 1 | Step 1 | N/A | User:Sreeniva_Team | N/A | Final Rejection |

For Check for Vacant:

The screenshot shows the Salesforce Setup interface with the 'Approval Processes' page open. The process is titled 'check for vacant' and is set to 'Active'. It includes fields for Unique Name ('check_for_vacant'), Description ('Tenant: check for vacant'), and Entry Criteria ('Deserted, status EQUALS Leaving'). The 'Next Automated Approver Determined By' field is set to 'Administrator ONLY'. Approval Assignment Email Template is 'Leave approved' and Initial Submitter is 'Tenant Owner'. The process was created by 'Sourya_Team' on 6/20/2015 at 3:18 AM and modified by 'Sourya_Team' on 6/26/2015 at 11:02 PM.

● Apex Trigger

Create an Apex Trigger

```

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

```

The screenshot shows the Salesforce Developer Console with an Apex trigger named 'testHandler' for the 'Tenant__c' object. The trigger logic prevents insertions for new records. A modal window is open, showing the 'Open' dialog with 'Triggers' selected in the Entity Type dropdown. The dialog lists a single entry named 'test'.

Developer Console - Google Chrome
orgfarm-5d1fe80592-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage

File Edit Debug Test Workspace Help < >

test.apex [testHandler.apex] MonthlytenantScheduler.apac []

Code Coverage Name API Version 41 Go To

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

Log Tests Checkpoints Query Editor View Status Progress Problems

None Use Problem

Create an Apex Handler class

Developer Console - Google Chrome
orgfarm-5d1fe80592-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage

File Edit Debug Test Workspace Help < >

test.apex [testHandler.apex] MonthlytenantScheduler.apac []

Code Coverage Name API Version 41 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) {
                newTenant.addError('A');
            }
        }
    }
}
```

Entity Type Entity Name Namespace Related

| Entity Type | Entity Name | Namespace | Related |
|------------------|-------------|------------------------|------------------------------------|
| Apex Trigger | testHandler | MonthlytenantScheduler | +< test ApexTrigger Referenced |
| Classes | | | +< property CustomField References |
| Triggers | | | +< Tenant__c Object References |
| Pages | | | +< Tenant__c Object References |
| Page Components | | | |
| Objects | | | |
| Static Resources | | | |
| Package | | | |

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

Log Tests Checkpoints Query Editor View Status Progress Problems

None Use Problem

Developer Console - Google Chrome

orgfarm-Sdf1e805f2-dev-ed.develop.my.salesforce.com/ui/common/apex/debug/ApexCSPage

testHandler.apc [testHandler.apc] HostedDynamicScheduler.apac []

Code Coverage Name: API Version: 64 Go To

```

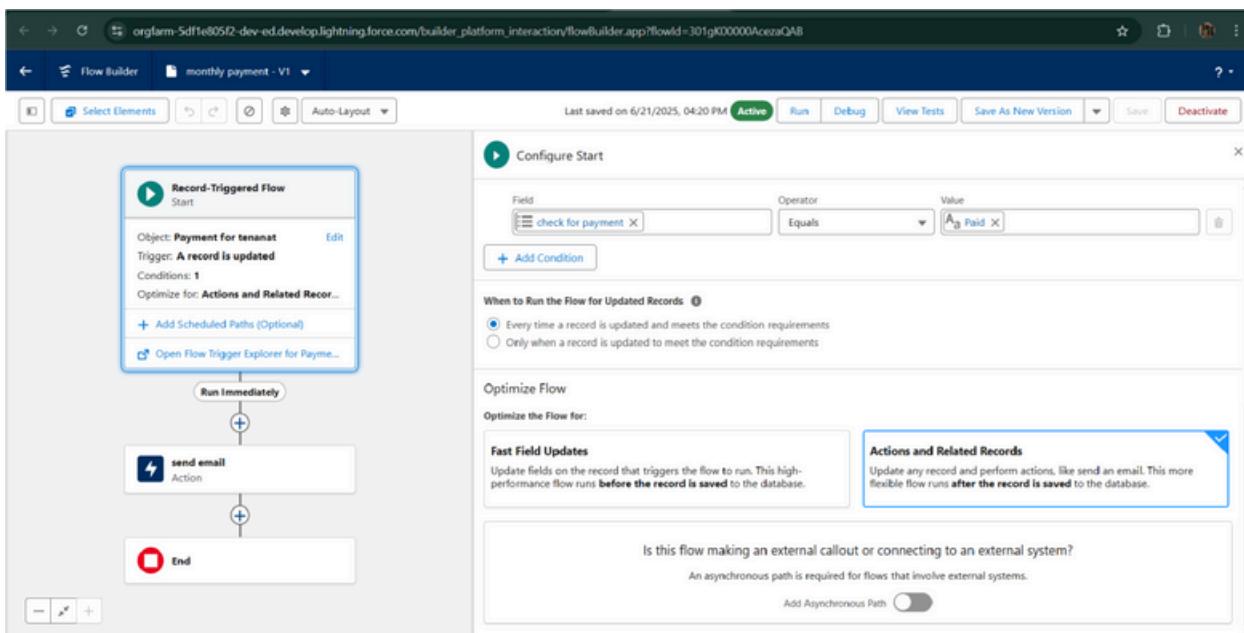
1  public class testHandler {
2
3    public static void preventInsert(List<Tenant__c> newList) {
4
5      Set<Id> existingPropertyIds = new Set<Id>();
6
7      for (Tenant__c existingTenant : [SELECT Id, Property__c FROM Tenant__c WHERE Property__c != null]) {
8
9        existingPropertyIds.add(existingTenant.Property__c);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

```

Log Test Checkpoints Query Editor View Status Progress Problems

Name Line Problem

● FLOWS



The screenshot shows the Salesforce Flow Builder interface with a flow titled "monthly payment - V1".

Flow Details:

- Start:** Record-Triggered Flow Start
- Object:** Payment for tenant
- Trigger:** A record is updated
- Conditions:** 1
- Optimize for:** Actions and Related Records
- Run Immediately:** Enabled
- Actions:** send email
- End:** End

Configuration Panel (Configure Start):

- Select Object:** Payment for tenant
- Configure Trigger:** Trigger the Flow When:
 - A record is created
 - A record is updated
 - A record is created or updated
 - A record is deleted
- Set Entry Conditions:** Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.
- Condition Requirements:** All Conditions Are Met (AND)

- Schedule class:
Create an Apex Class

```
Developer Console - Google Chrome
File * Edit * Debug * Test * Workspace * Help < >
InfoLog [ ] MonthlyEmailScheduler.apc [ ]
Code Coverage: None * API Version: 54 [ ]
1 * global class MonthlyEmailScheduler implements Schedulable {
2
3 *     global void execute(SchedulableContext sc) {
4
5         Integer currentDay = Date.today().day();
6
7         if (currentDay == 1) {
8
9             sendMonthlyEmails();
10
11     }
12
13 }
14
15
16 *     public static void sendMonthlyEmail
17
18     List<Tenant__c> tenants = [SEL
19
20         for (Tenant__c tenant : tenants) Open Filter Filter the repository (+ any string) Hide Managed Packages Refresh
21
22             String recipientEmail = tenant.Email__c;
23
Logs * Tests * Checklists * Query Editor * View Status * Progress * Problems
Name Line Problem
```

```

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             sendMonthlyEmails();
9         }
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 emailSubject = 'Please pay your monthly rent';
25
26         String emailContent = 'I trust this email finds you well. I am writing to remind you that the monthly rent is due now. Timely payment ensures the smooth functioning of our rental arrangement and helps maintain a positive living environment for all.' ;
27
28         Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
29
30         mail.setTo(recipientEmail);
31         mail.setSubject(emailSubject);
32         mail.setPlainTextEmailBody(emailContent);
33
34         messaging.sendEmail(new Messaging.SingleEmailMessage[]{mail});
35
36     }
37
38 }
39
40 }

```

Schedule Apex class

The screenshot shows the Salesforce Setup interface with the search bar set to "apex". The left sidebar is expanded to show categories like Email, Custom Code, and Environments. Under "Apex Classes", the "Apex Classes" tab is selected, displaying the details for the "MonthlyEmailScheduler" class.

| Apex Class Detail | |
|-------------------|-----------------------|
| Name | MonthlyEmailScheduler |
| Namespace Prefix | |
| Created By | Scomya Team |
| Status | Active |
| Code Coverage | 0% (0/15) |
| Last Modified By | Scomya Team |

The "Class Body" tab is selected, showing 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             sendMonthlyEmails();
9         }
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 emailSubject = 'Please pay your monthly rent';
25
26         String emailContent = 'I trust this email finds you well. I am writing to remind you that the monthly rent is due now. Timely payment ensures the smooth functioning of our rental arrangement and helps maintain a positive living environment for all.' ;
27
28         Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
29
30         mail.setTo(recipientEmail);
31         mail.setSubject(emailSubject);
32         mail.setPlainTextEmailBody(emailContent);
33
34         messaging.sendEmail(new Messaging.SingleEmailMessage[]{mail});
35
36     }
37
38 }
39
40 }

```

Lease Management

Tenant Aswini

Related Details

* = Required Information

Tenant Name: Aswini

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

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

Tenant Aswini

Related Details

* = Required Information

Tenant Name: Aswini

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

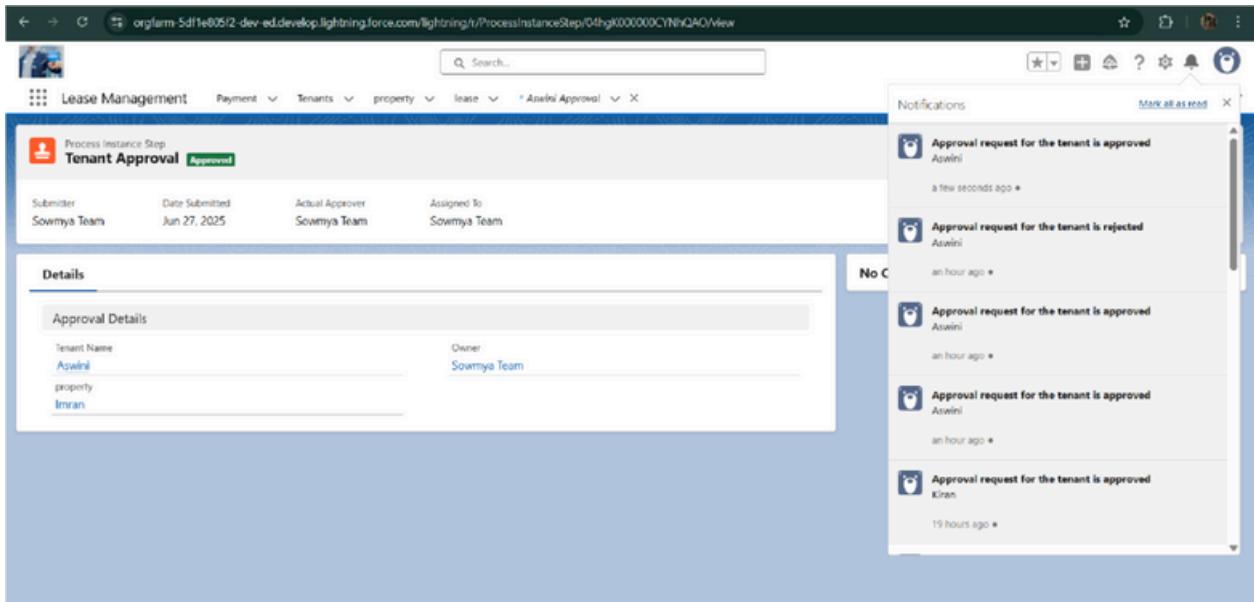
Activity

Upcoming & Overdue

No activities to show.

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

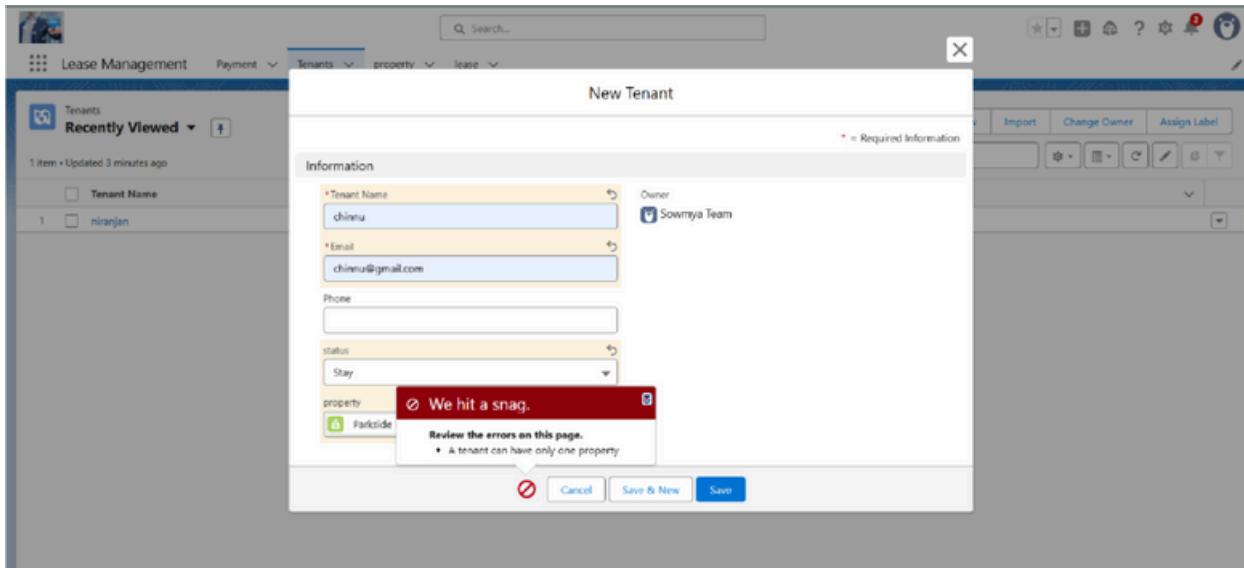
No past activity. Past meetings and tasks marked as done show up here.



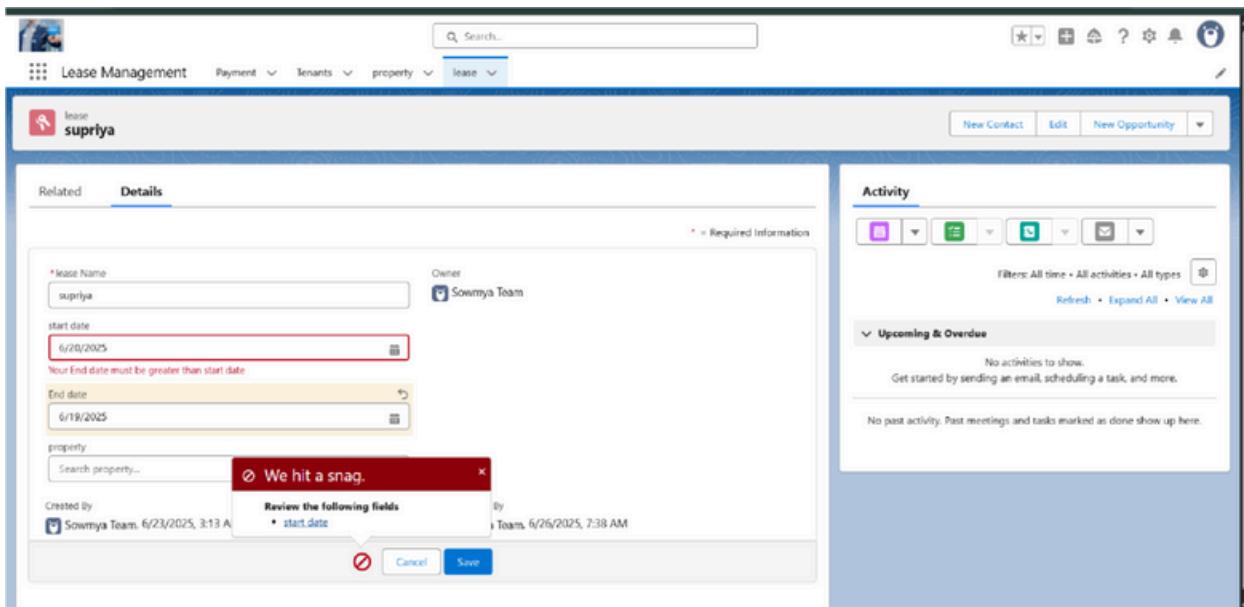
FUNCTIONAL AND PERFORMANCE TESTING

Performance Testing

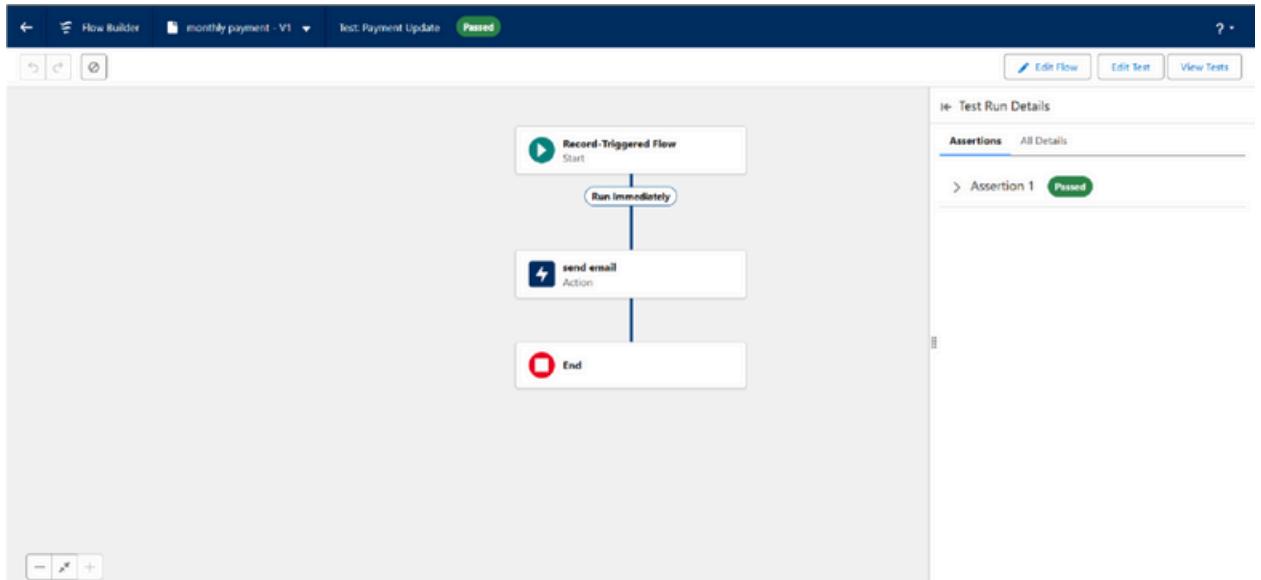
- Trigger validation by entering duplicate tenant-property records



● Validation Rule checking



- Test flows on payment update



- Approval process validated through email alerts and status updates

The screenshot shows a CRM interface for 'Lease Management'. On the left, a 'Tenant' record for 'niranjan' is displayed with tabs for 'Related' and 'Details'. The 'Details' tab is active, showing fields for Tenant Name (niranjan), Email (niranjan1506@gmail.com), Phone, Status (Stay), and Property (Parkside Lofts). It also shows creation and last modified details by 'Sowmya Team'. On the right, a 'Notifications' sidebar 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
- 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 a CRM interface for 'Lease Management'. On the left, a 'Tenant' record for 'niranjan' is displayed with tabs for 'Related' and 'Details'. The 'Details' tab is active, showing sections for 'Approval History (6+)', 'Payment (2)', and 'View All'. The 'Approval History' section lists six steps: Step 1 (Approved, Sowmya Team, 6/25/2025, 5:39 AM), Approval Request Submitted (Submitted, Sowmya Team, 6/25/2025, 5:39 AM), Step 1 (Rejected, Sowmya Team, 6/23/2025, 3:59 AM), Approval Request Submitted (Submitted, Sowmya Team, 6/23/2025, 3:58 AM), Step 1 (Approved, Sowmya Team, 6/23/2025, 3:55 AM), and Approval Request Submitted (Submitted, Sowmya Team, 6/23/2025, 3:55 AM). The 'Payment' section shows two entries: 'Jack' and 'Rahul'. On the right, a sidebar displays a message: '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

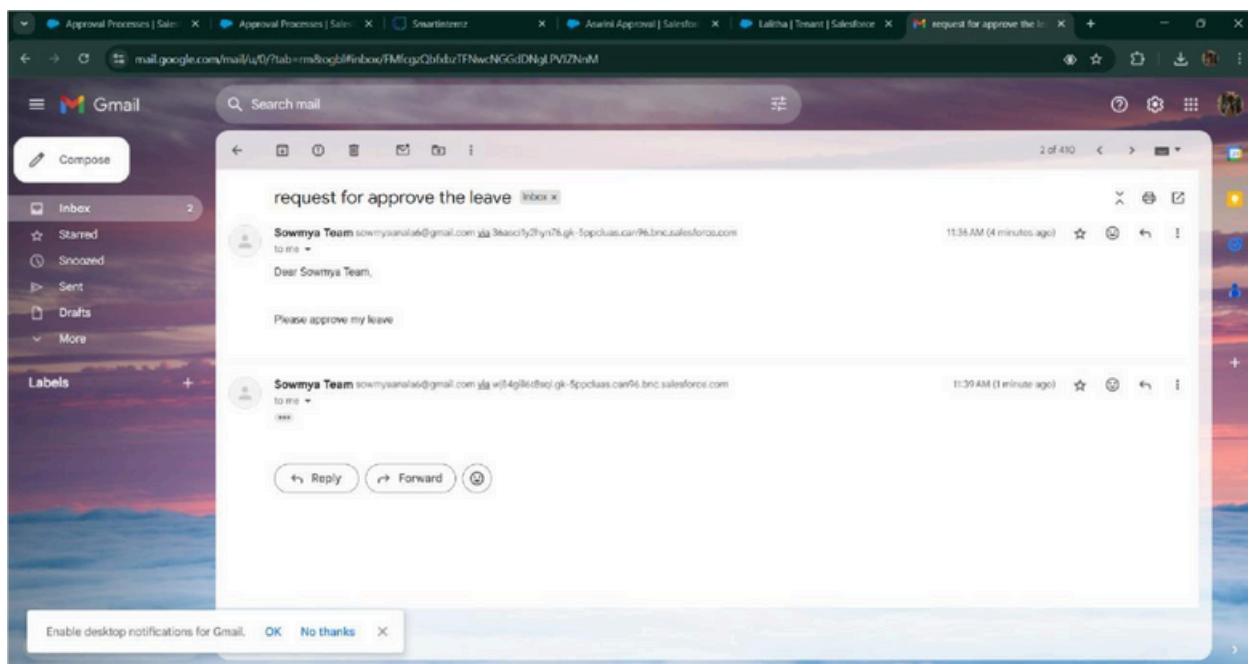
The screenshot shows the Salesforce Setup interface under the 'Custom Tabs' section. It displays three main sections: 'Custom Object Tabs', 'Web Tabs', and 'Visualforce Tabs'. The 'Custom Object Tabs' section lists tabs for 'Issue', 'Payment', 'Record', and 'Records', each with a 'Tab Style' icon (e.g., Keys, Credit card, Back, Map) and a 'Description' column. The 'Web Tabs' and 'Visualforce Tabs' sections both indicate 'No Web Tabs have been defined' and 'No Visualforce Tabs have been defined'.

● Email alerts

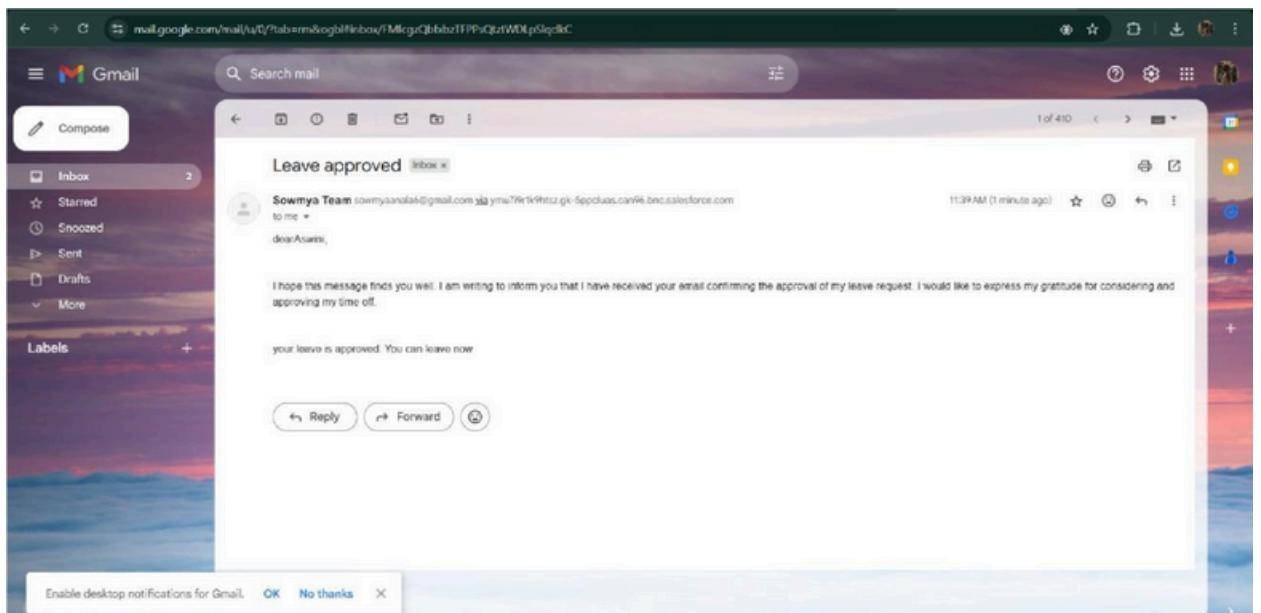
The screenshot shows the 'Lease Management' application interface. The top navigation bar includes 'Lease Management', 'Payment', 'Tenants', 'property', and 'lease'. Below the navigation is a search bar and a toolbar with various icons. The main area displays a grid titled 'Approval History' with 8 items. The grid columns are: Step Name, Date, Status, Assigned To, Actual Approver, and Comments. The data in the grid is as follows:

| Step Name | Date | Status | Assigned To | Actual Approver | Comments |
|------------------------------|--------------------|-----------|-------------|-----------------|-------------------|
| 1 Step 1 | 6/23/2025, 5:39 AM | Approved | Sowmya Team | Sowmya Team | approved |
| 2 Approval Request Submitted | 6/23/2025, 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 |

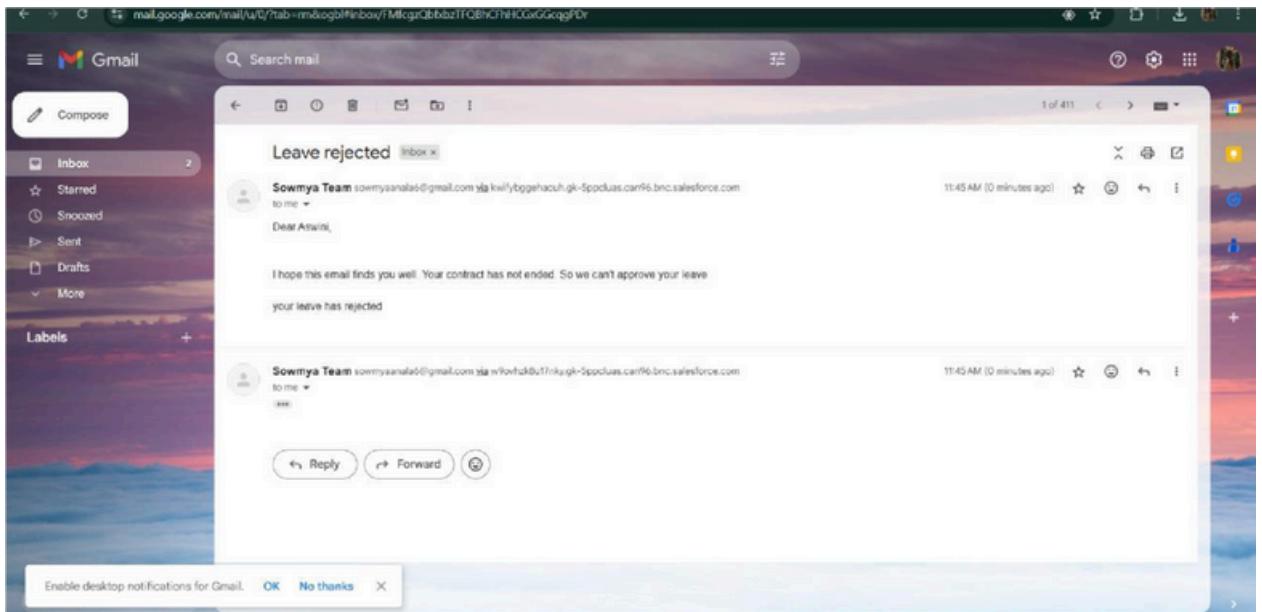
● Request for approve the leave



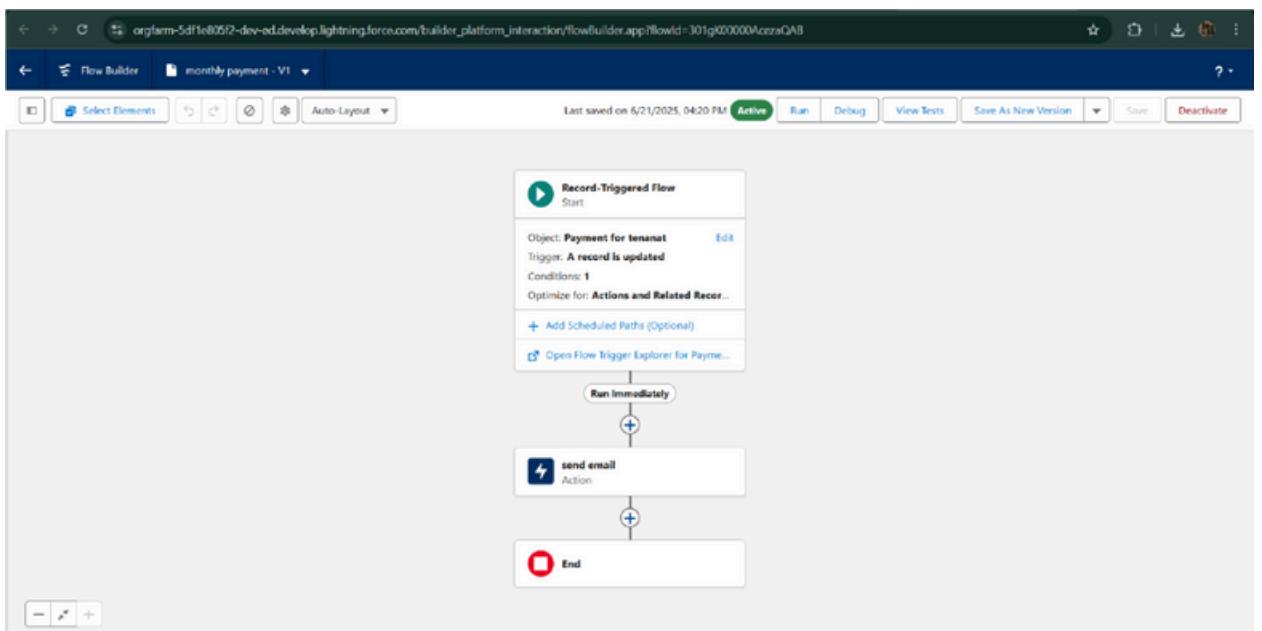
- Leave approved



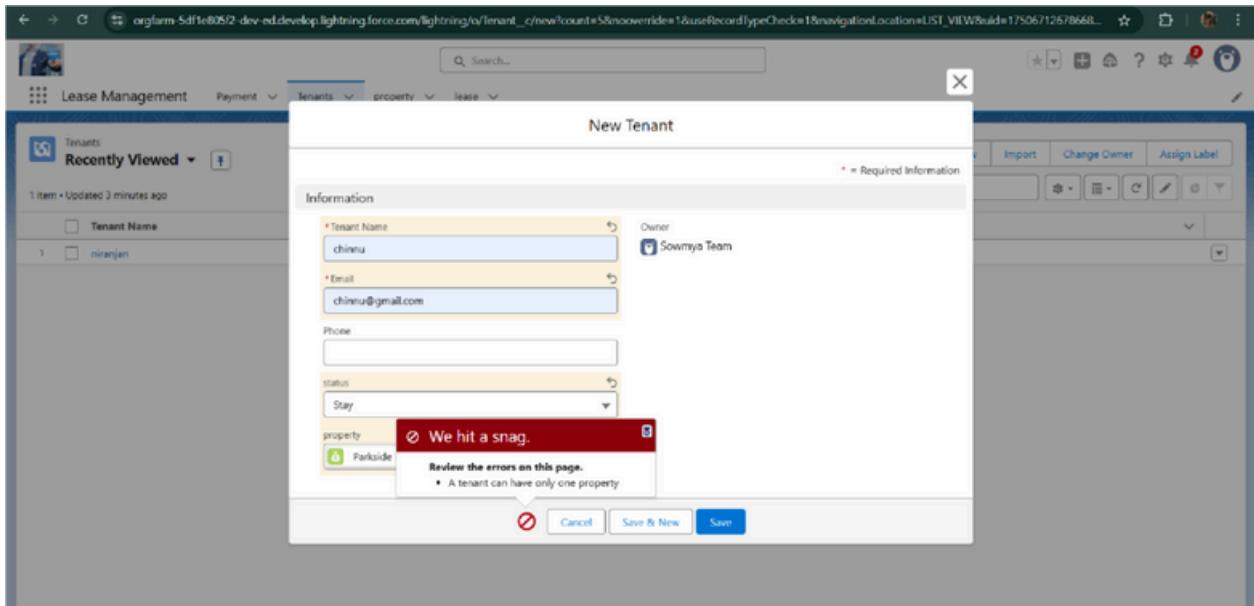
- Leave rejected



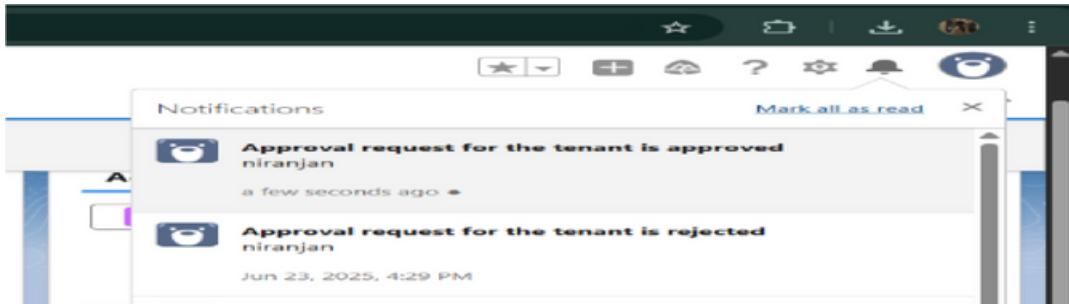
● Flow runs



● Trigger error messages



● Approval process notifications



ADVANTAGES & DISADVANTAGES

Advantages

1. Cost Predictability

- Lease agreements typically involve fixed payments, making budgeting and forecasting easier.

2. Access to High-Value Assets

- Allows businesses to use expensive assets (e.g., real estate, machinery, vehicles) without large upfront capital investment.

3. Flexibility

- Leasing can offer flexibility in contract length, upgrades, and expansion options.

4. Tax Benefits

- Lease payments can often be deducted as business expenses, reducing taxable income.

5. Reduced Maintenance Responsibility

- Depending on the lease type, maintenance and repairs may be the responsibility of the lessor (e.g., in triple net or full-service leases).

Disadvantages

1. Long-Term Cost

- Over time, leasing can be more expensive than purchasing the asset outright.

2. Limited Control

- Leased assets are not owned, so modifications or usage may be restricted by lease terms.

3. Contractual Obligations

- Lease agreements may lock businesses into long-term commitments, limiting flexibility if business needs change.

4. Complexity

- Managing multiple lease agreements across locations or asset types can be administratively complex without the right system.

5. Penalties

- Early termination, overuse, or violations of lease terms can result in penalties or legal disputes.

CONCLUSION

TheLease Management System successfully streamlines the operations of leasing through a structured, automatedSalesforce 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});  
    }  
}  
}
```