

Lease Management

College Name: Kathir college of arts and science

College Code: bruah

TEAM ID:

TEAM MEMBERS:

Team LeaderName: ABINAYA B

Email: qweendevil04@gmail.com

Team Member: DEVI ANNAPOORNA F

Email: deviannapoorna155@gmail.com

Team Member: LAVANYA M(04-03-2006)

Email: lavanyamuruganantham2006@gmail.com

Team Member: NIVETHA P

Email: nivethadhanamani1325@gmail.com

Team Member: GOWRI K

Email: gowri28092005@gmail.com

1. INTRODUCTION

1.1 Project Overview

The Lease Management System is a Salesforce-based application designed to streamline the processes associated with leasing real estate properties. It handles tenant management, lease contracts, payments, and communication with automation features such as flows, approval processes, and email alerts.



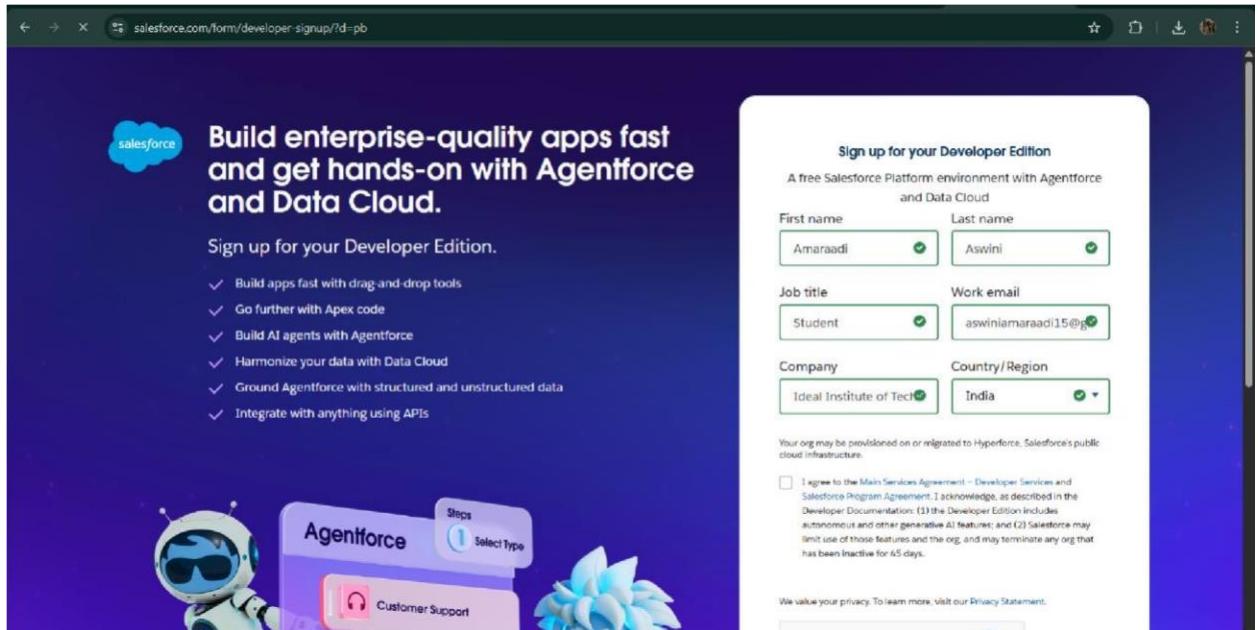
1.2 Purpose

The main objective of the project is to enable organizations to efficiently manage properties, tenants, and lease-related activities. It reduces manual intervention, improves accuracy, and ensures better compliance and communication.

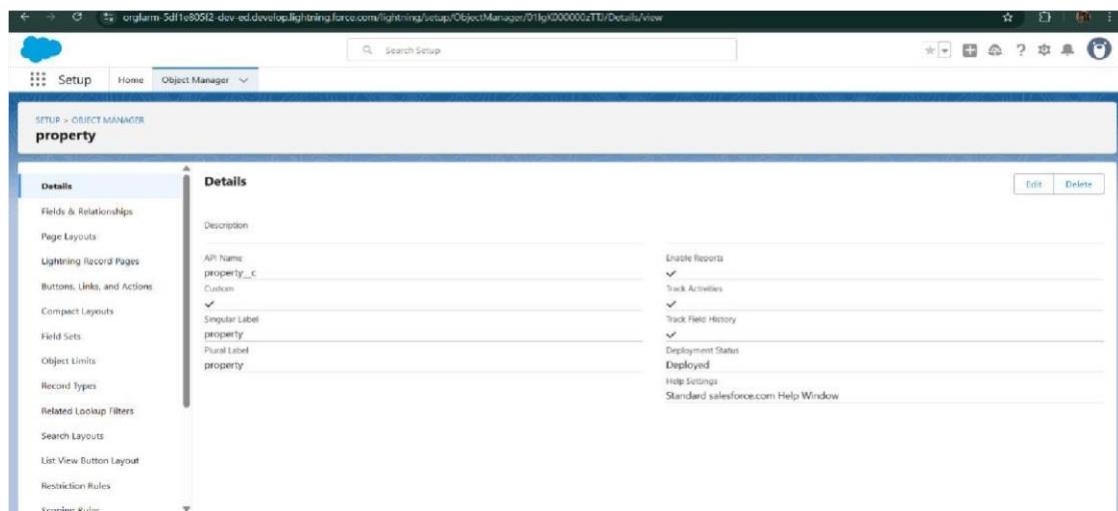
DEVELOPMENT PHASE

Creating Developer Account:

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



- Created objects: Property, Tenant, Lease, Payment



SETUP > OBJECT MANAGER

Tenant

Details

Description

API Name: Tenant__c

Custom: ✓

Singular Label: Tenant

Plural Label: Tenants

Enable Reports: ✓

Track Activities: ✓

Track Field History: ✓

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

Scoping Rules

SETUP > OBJECT MANAGER

lease

Details

Description

API Name: lease__c

Custom: ✓

Singular Label: lease

Plural Label: lease

Enable Reports: ✓

Track Activities: ✓

Track Field History: ✓

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

SETUP > OBJECT MANAGER

Payment for tenant

Details

Description

API Name: Payment_for_tenant__c
Custom: ✓
Singular Label: Payment for tenant
Plural Label: Payment

Enable Reports: ✓
Track Activities: ✓
Track Field History: ✓
Deployment Status: Deployed
Help Settings: Standard salesforce.com Help Window

Fields & Relationships

- Page Layouts
- Lightning Record Pages
- Buttons, Links, and Actions
- Compact Layouts
- Field Sets
- Object Limits
- Record Types
- Related Lookup Filters
- Search Layouts
- List View Button Layout
- Restriction Rules
- Scoping Rules

- Configured fields and relationships

SETUP > OBJECT MANAGER

property

Fields & Relationships

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)	✓	
sflq	sflq__c	Text(18)		
Type	Type__c	Picklist		

Fields & Relationships

- Page Layouts
- Lightning Record Pages
- Buttons, Links, and Actions
- Compact Layouts
- Field Sets
- Object Limits
- Record Types
- Related Lookup Filters
- Search Layouts
- List View Button Layout
- Restriction Rules
- Scoping Rules

Setup > Object Manager
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	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				
7 Items, Sorted by Field Label				
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		

Fields & Relationships
7 items. Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Email	Email_c	Email		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Phone	Phone_c	Phone		
status	status_c	Picklist		
Tenant Name	Name	Text(80)		✓

- Developed Lightning App with relevant tabs

App Details & Branding

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

App Details

* App Name: Lease Management

* Developer Name: Lease_Management

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

App Branding

Image:

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

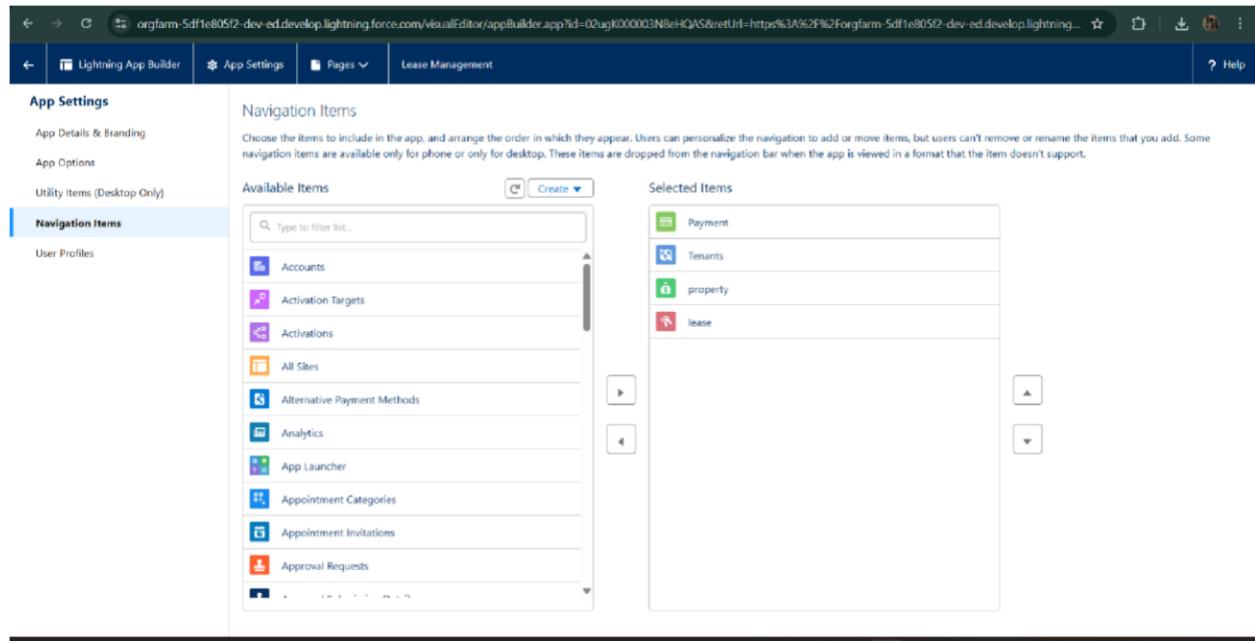
Available Items

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.

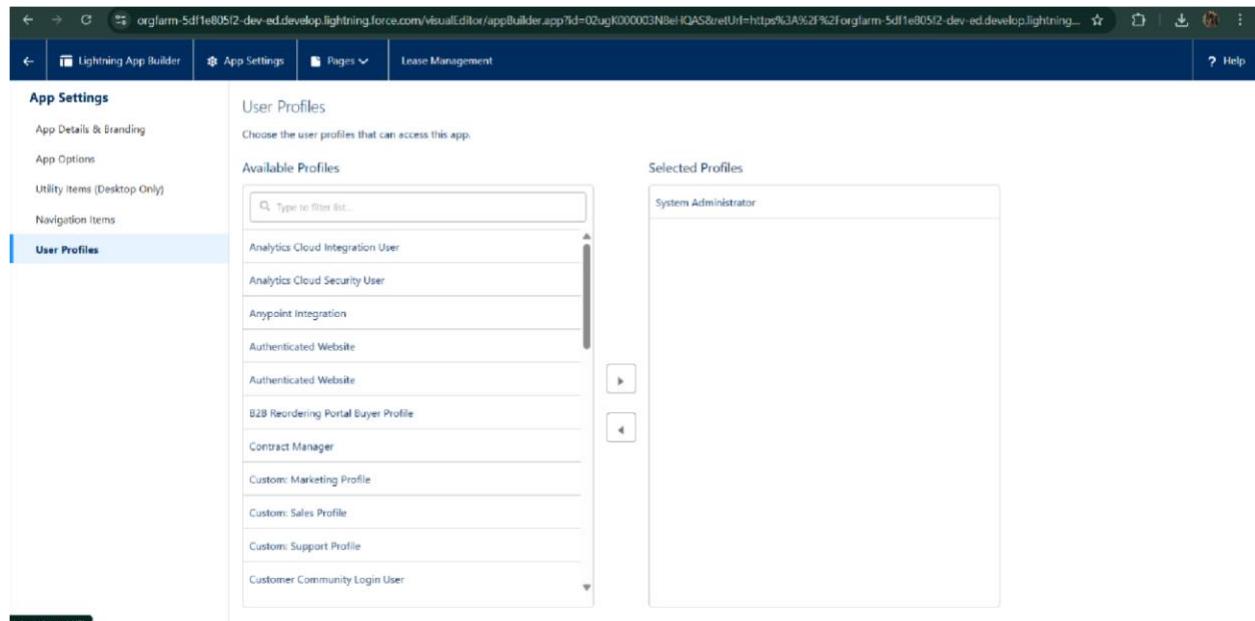
Available Profiles

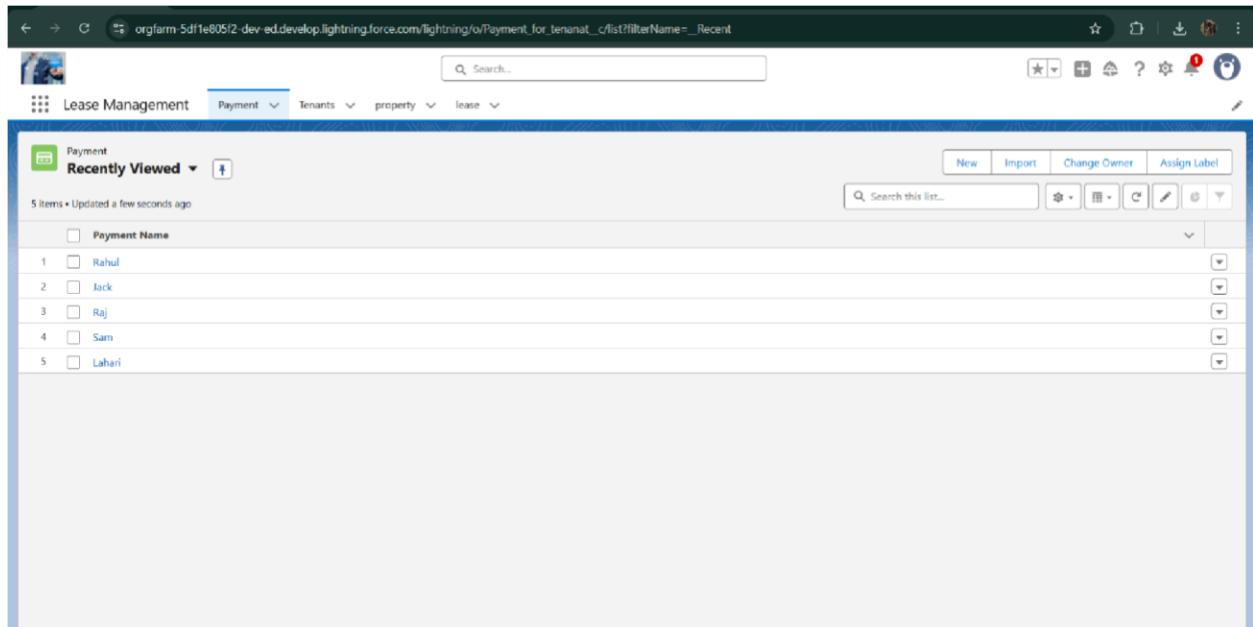
Type to filter list...

- 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

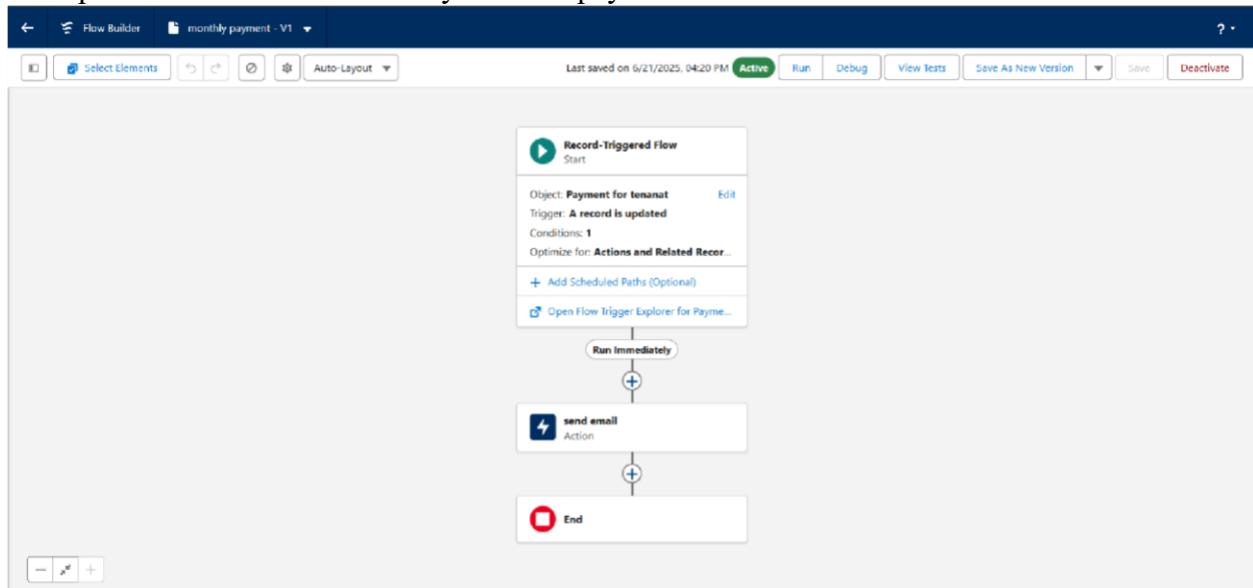
Selected Profiles

- System Administrator





- Implemented Flows for monthly rent and payment success

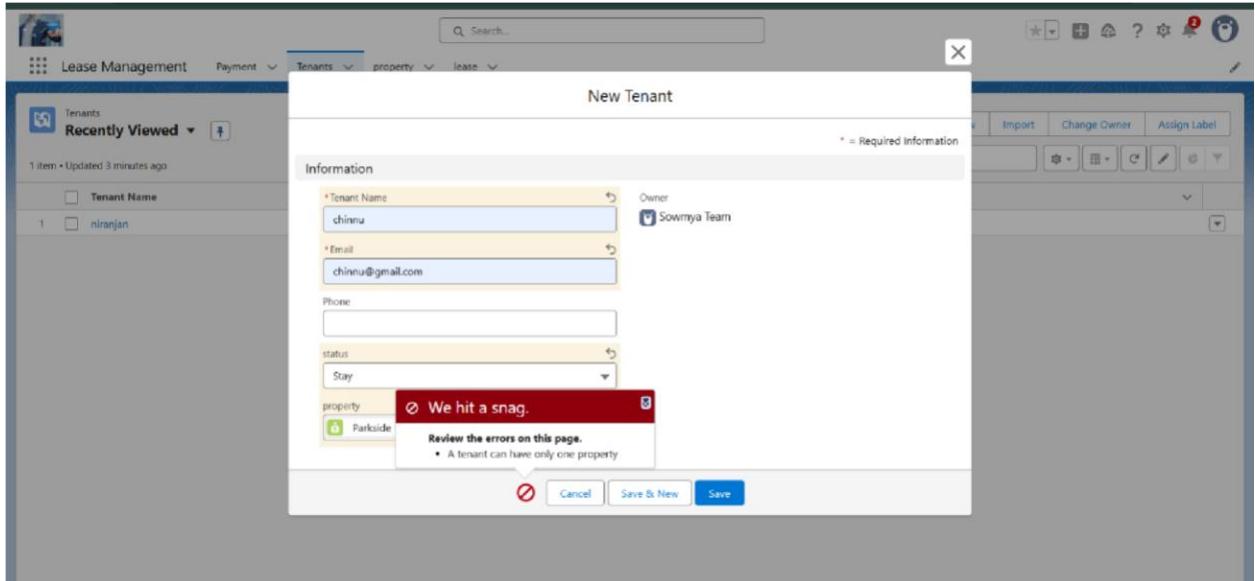


- To create a validation rule to a Lease Object

The screenshot shows the 'Validation Rule Edit' screen for the 'lease' object. The 'Rule Name' is set to 'lease_end_date'. The 'Active' checkbox is checked. The 'Error Condition Formula' field contains the formula 'End_date_c <= start_date_c'. A tooltip for the 'ABS' function is displayed, stating: 'Returns the absolute value of a number, a number without its sign'. The 'Functions' dropdown menu lists various mathematical and string functions.

The screenshot shows the 'Validation Rule Detail' screen for the 'lease Validation Rule'. The rule name is 'lease_end_date'. The error condition formula is 'End_date_c <= start_date_c'. The error message is 'Your End date must be greater than start date'. The error location is 'start date'. The rule was created by 'Sowmya Team' on 6/19/2025 at 5:37 AM and modified by 'Sowmya Team' on 6/26/2025 at 7:47 AM.

- Added Apex trigger to restrict multiple tenants per property



- Scheduled monthly reminder emails using Apex class

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

- File → Edit → Debug → Test → Navigation → Help**
- Test Page**: `test.page` | `testMonthlyEmailScheduler.apex` | `MonthlyEmailScheduler.apex`
- Code Coverage**: New | API Version: 44
- Apex Class Content (MonthlyEmailScheduler):**

```
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 }
```

- Log**: Tasks | Checkpoints | Query Editor | View Status | Response | Problems
- New**: Use | Politeness

- 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: Somanya Team [Change]
 - Description: Created By: Somanya Team, 6/20/2025, 1:08 AM
 - Modified By: Somanya Team, 6/20/2025, 1:08 AM
- 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: Somanya Team [Change]
 - Description: Created By: Somanya Team, 6/20/2025, 1:06 AM
 - Modified By: Somanya 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

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	Sonamya_Team [Change]
Description	Created By Sonamya_Team, 6/20/2025, 1:11 AM
Created By	Sonamya_Team, 6/20/2025, 1:11 AM
Modified By	Sonamya_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	Sonamya_Team [Change]
Description	Created By Sonamya_Team, 6/20/2025, 1:12 AM
Created By	Sonamya_Team, 6/20/2025, 1:12 AM
Modified By	Sonamya_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 search bar set to "email template". The "Email" section is expanded, showing "Classic Email Templates" selected. A specific template, "tenant payment", is highlighted. 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 template content:

```

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". The "Data" section is expanded, showing "Approval Processes" selected. A specific process, "Tenant: TenantApproval", is highlighted. The "Process Definition Detail" section shows the following information:

- Process Name: TenantApproval
- Unique Name: TenantApproval
- Description:
- Entry Criteria: `TENANT__C.STATUS EQUALS Stay`
- Record Eligibility: `Administrator ONLY`
- Next Automated Approver Determined By:
- Allow Submitters to Recall Approval Requests:
- 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:

Approval Processes

Tenant: check for vacant

Process Definition Detail

Process Name:	check for vacant
Unique Name:	check_for_vacant
Description:	Tenant: check for vacant
Entry Criteria:	Tenant: status EQUALS Leaving
Requestor Identity:	Administrator ONLY
Approval Assignment Email Template:	Leave approved
Initial Submitters:	Tenant Owner
Created By:	Sowmya_Team
Modified By:	Sowmya_Team

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	step1			User:Sowmya_Team	Final Rejection

- Apex Trigger

Create an Apex Trigger

```

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

```

Open

Entity Type	Edition	Related
Classes	Name	Namespace
Triggers	test	
Pages		
Page Components		
Objects		
Static Resources		
Packages		

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/ApexCSPage. The tab bar shows `testApex`, `testHandler.apex`, and `MonthlyEmailScheduler.apex`. The code editor contains the following Apex trigger:

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

Create an Apex Handler class

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/ApexCSPage. The tab bar shows `testApex`, `testHandler.apex`, and `MonthlyEmailScheduler.apex`. The code editor contains the following Apex 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 t');
            }
        }
    }
}
```

A modal window titled "Open" is displayed, showing the relationships of the `testHandler` class. The table lists:

Entity Type	Status	Related
Classes	testHandler	Name Extent Direction
Triggers	MonthlyEmailScheduler	← test ApexTrigger Referenced
Pages		← property CustomField References
Components		← Tenant__c SObject References
Objects		← Tenant__c SObject References
Static Resources		
Package		

Developer Console - Google Chrome

orgfarm-5df1e805f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage

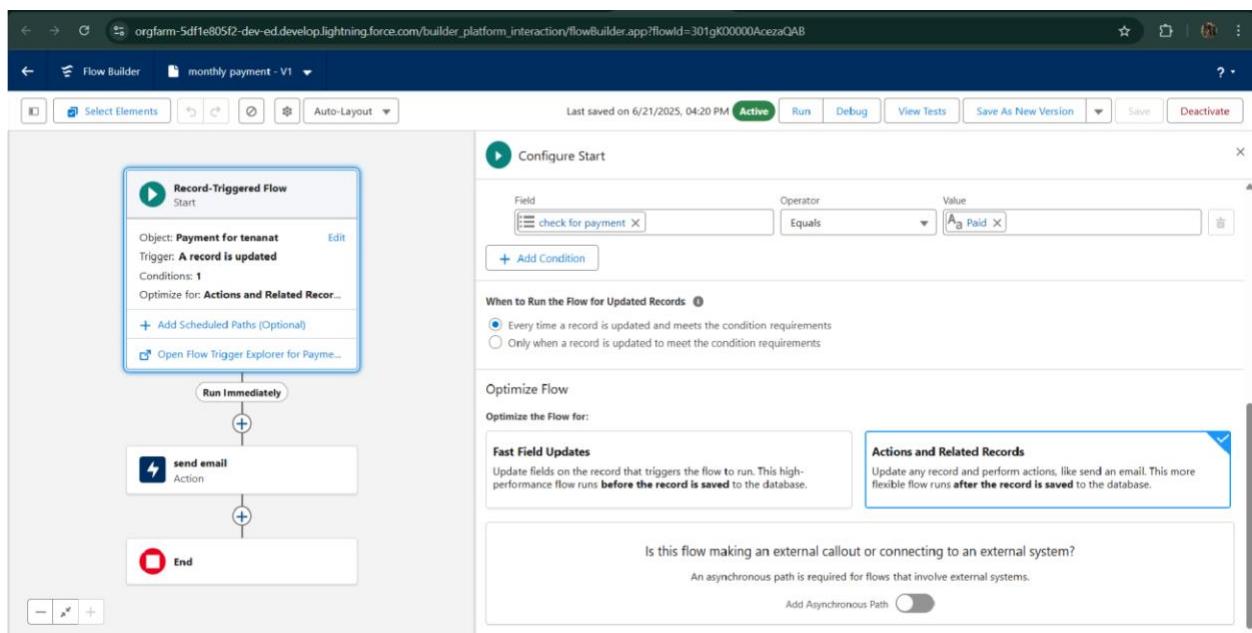
File Edit Debug Test Workspace Help < >

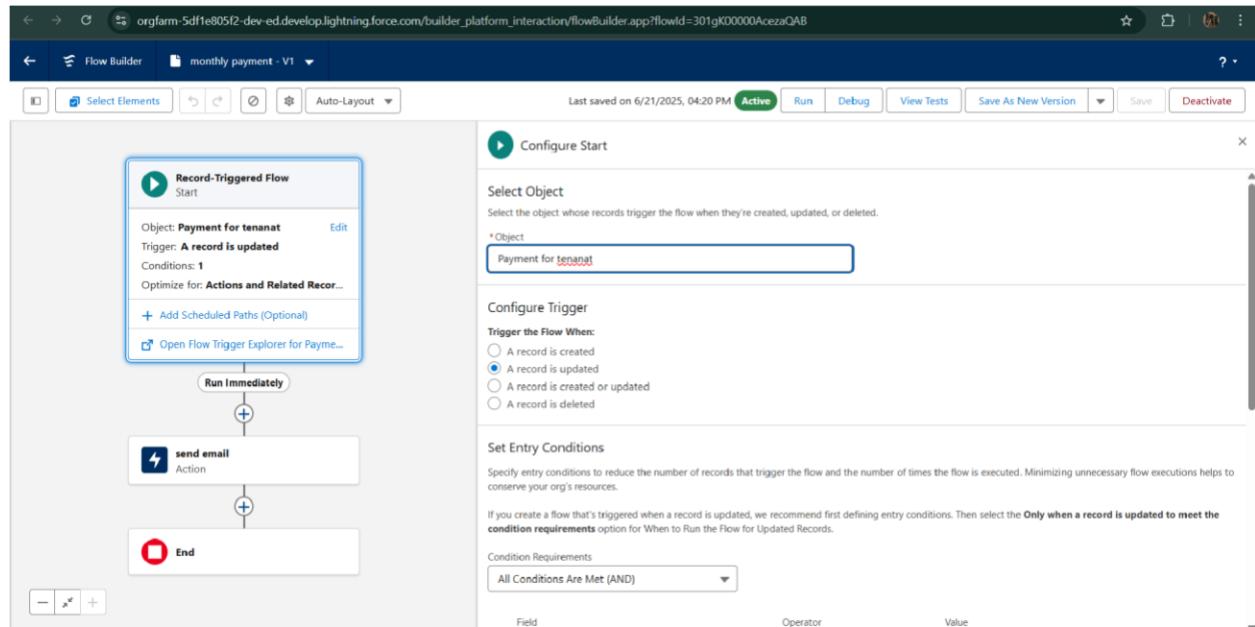
testHandler.apc MonthlyEmailScheduler.apc * []

Code Coverage: None API Version: 54 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                 newTenant.addError('A tenant can have only one property');
20
21             }
22
23         }
24     }
25 }
```

- FLOWS





- Schedule class:
Create an Apex Class

The screenshot shows the Salesforce Developer Console. The code editor displays the Apex class "MonthlyEmailScheduler". The class implements the "Scheduleable" interface and contains a scheduled method "execute" that sends monthly emails to tenants if the current day is 1. A modal window titled "Open" is overlaid on the console, showing the repository browser with the "Classes" tab selected. The "MonthlyEmailScheduler" class is listed under the "Name" column.

```

1 * global class MonthlyEmailScheduler implements Scheduleable {
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 = [SELE
19
20     for (Tenant__c tenant : tenants
21
22         String recipientEmail = tenant.Email__c;
23

```

```

1. *@ISOBEL 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.setToRecipients(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 Apex Classes page. The sidebar on the left has a search bar and navigation links for Email, Custom Code, Environments, and Jobs. The main area displays the Apex Class Detail for 'MonthlyEmailScheduler'. The 'Class Body' tab is selected, showing the Apex code:

```

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.setToRecipients(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. }

```

The class details table shows the following information:

Name	Namespace Prefix	Status	Active
MonthlyEmailScheduler		Code Coverage	0% (0/15)
		Last Modified By	Somya_Team , 6/23/2025, 2:47 AM

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

Cancel Save javascript:void(0)

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 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

Cancel Save

Tenant was submitted for approval.

New Contact Edit New Opportunity

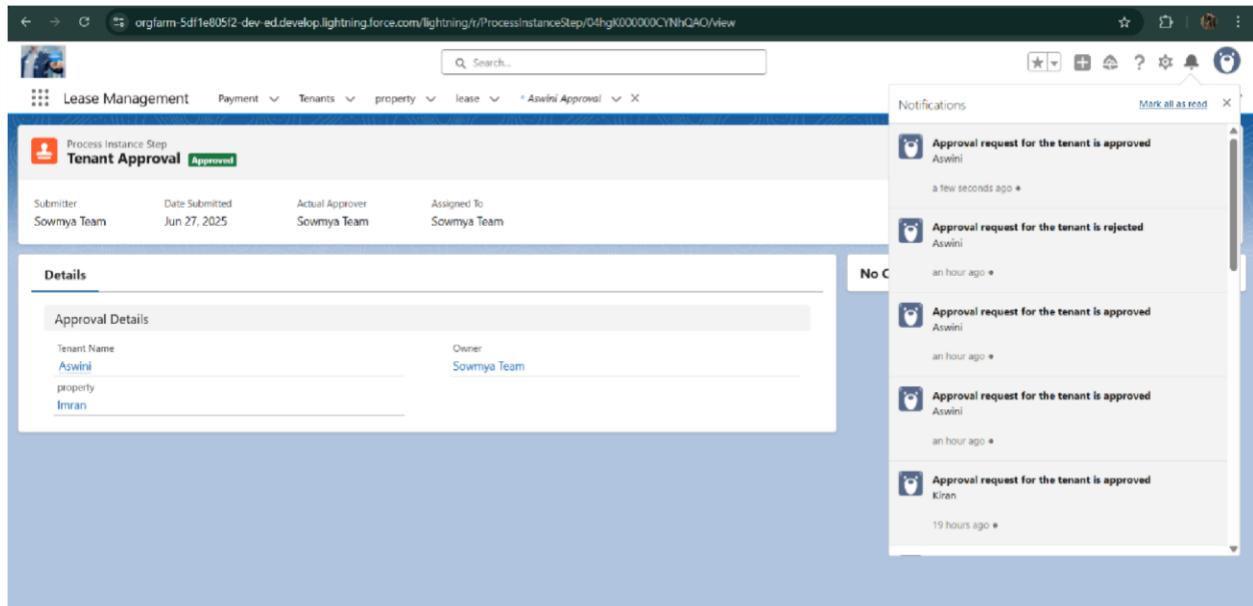
Activity

Filters: All time • All activities • All types 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.

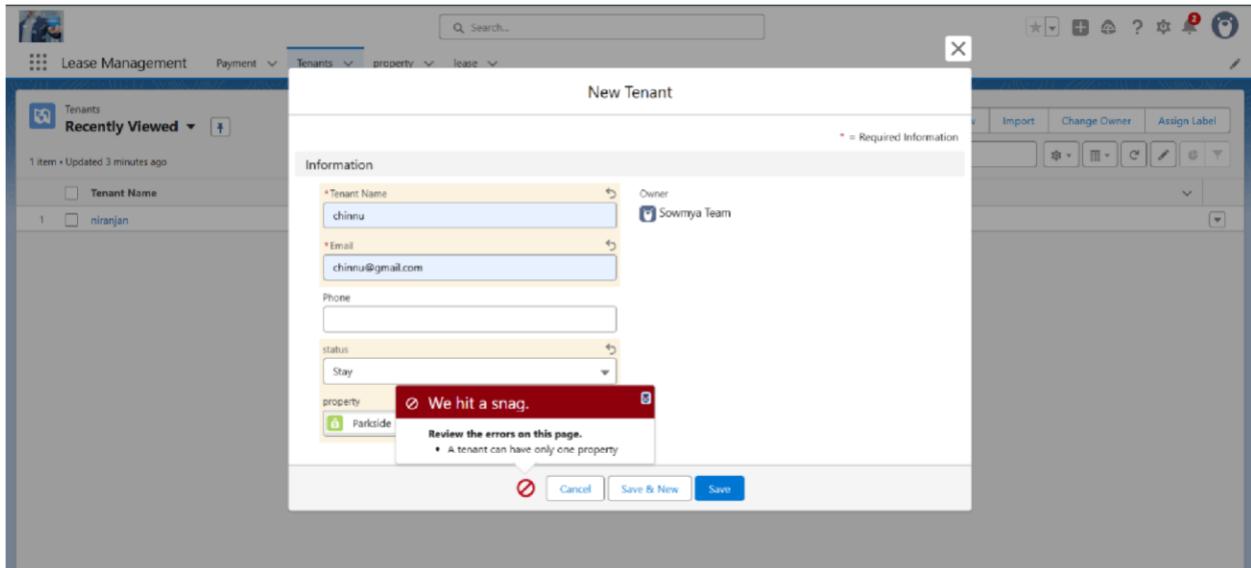
30°C Cloudy Search ENG IN 12:46 27-06-2025



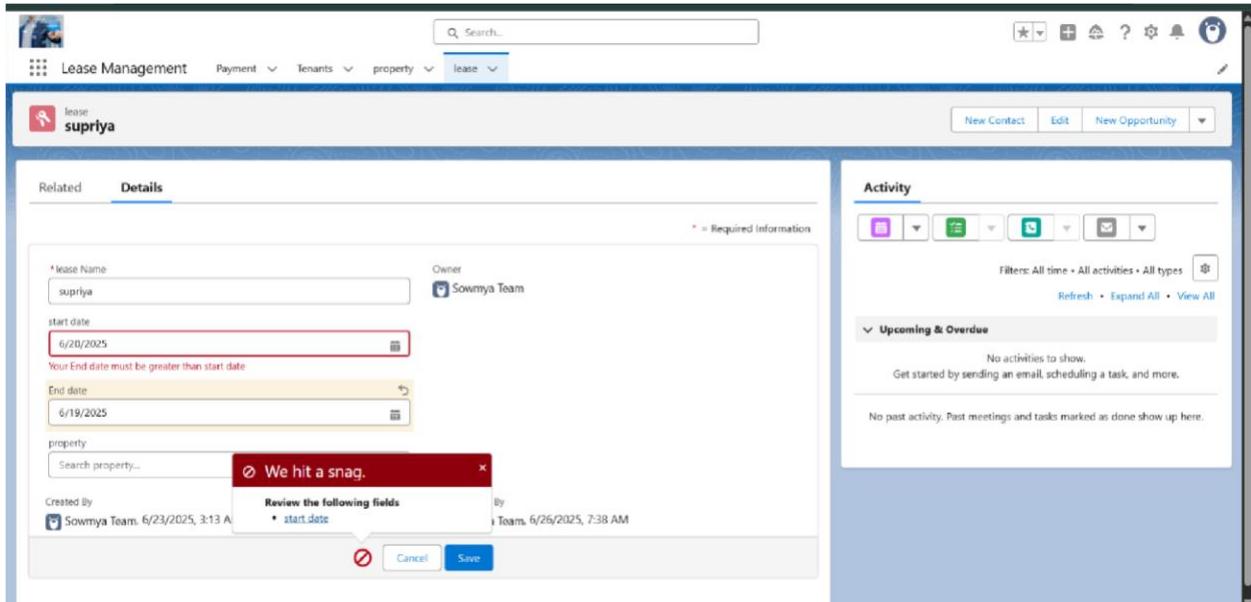
FUNCTIONAL AND PERFORMANCE TESTING

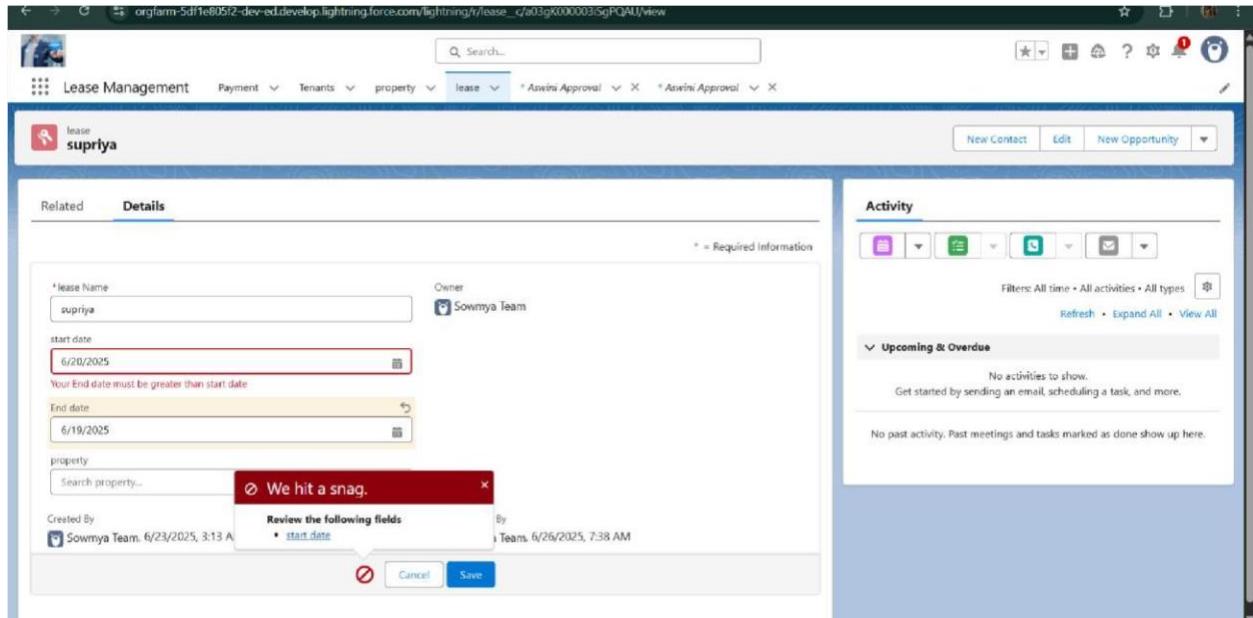
Performance Testing

- Trigger validation by entering duplicate tenant-property records

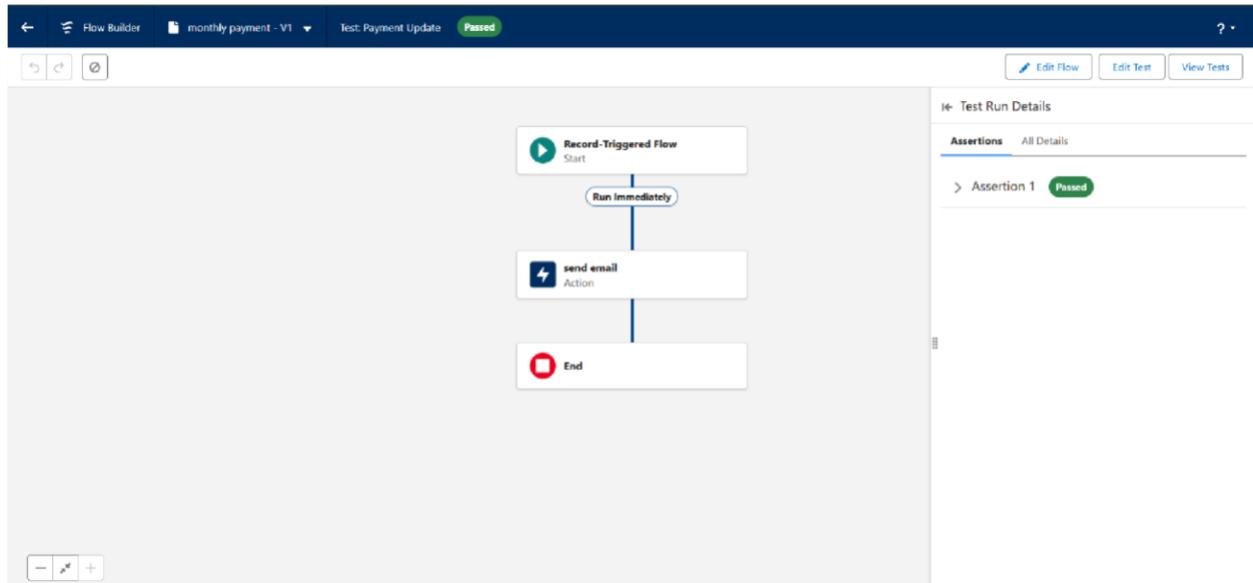


- Validation Rule checking





- Test flows on payment update



- Approval process validated through email alerts and status updates

The screenshot shows the 'Lease Management' application interface. The top navigation bar includes 'Lease Management', 'Payment', 'Tenants', 'property', 'lease', and 'niranjan Approval'. The main content area displays the 'Details' tab for tenant 'niranjan'. The form fields include:

- Tenant Name: niranjan
- Email: niranjan1506@gmail.com
- Phone: (empty)
- Status: Stay
- Property: Parkside Lofts
- Created By: Sowmya Team (6/23/2025, 2:33 AM)
- Last Modified By: Sowmya Team (6/23/2025, 3:58 AM)

Buttons at the bottom are 'Cancel' and 'Save'.

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)
- New Guidance Center learning resource available: Define Your Sales Process (Jun 20, 2025, 1:28 PM)

Mark all as read

The screenshot shows the same 'Lease Management' application interface for tenant 'niranjan'. The 'Approval History' section is expanded, displaying the following table:

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

A 'View All' button is located below the table.

The 'Payment' section is also expanded, showing:

Payment Name
Jack
Rahul

A 'New' button is located to the right of the payment list. A message bar at the top right 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

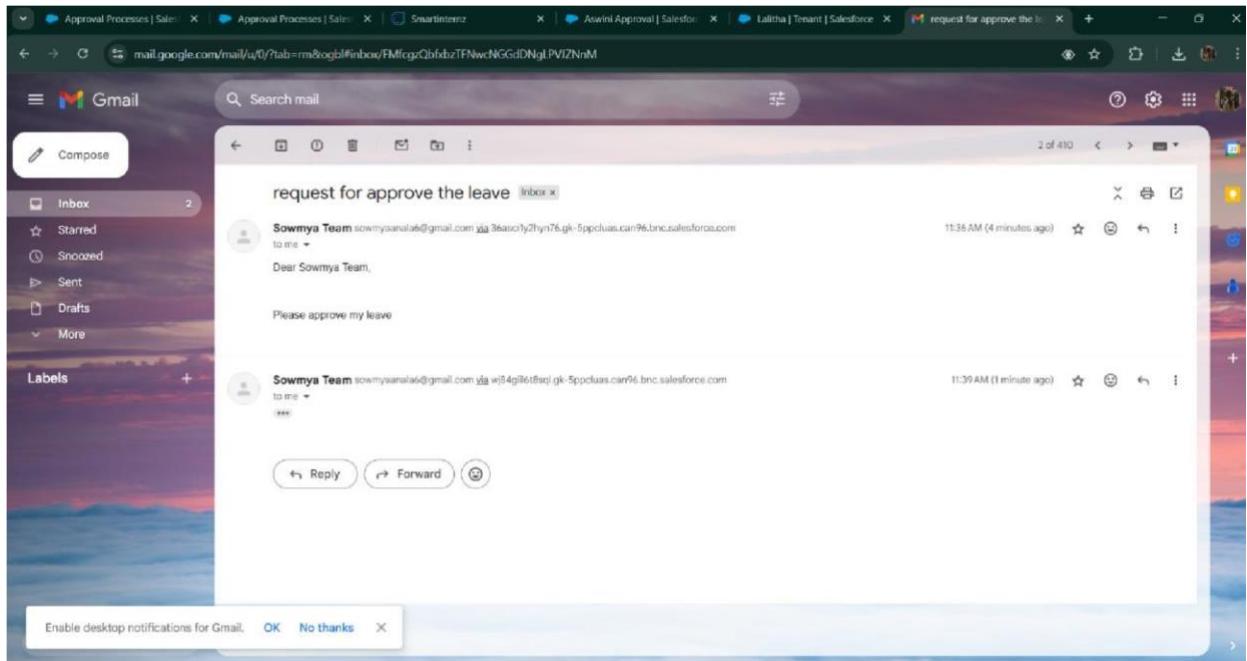
The screenshot shows the Salesforce Setup interface under the 'Custom Tabs' section. It displays a table of custom object tabs with columns for Action, Label, Tab Style, and Description. The tabs listed are Lease (Style: Keys), Payment (Style: Credit card), Property (Style: Back), and Tenant (Style: Map). A note at the top states: "You can create new custom tabs to extend Salesforce functionality or to build new application functionality. Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external web applications and content within the Salesforce window. Visualforce tabs allow you to embed Visualforce pages. Lightning Component tabs allow you to add Lightning components to the navigation menu in Lightning Experience and the mobile app. Lightning Page tabs allow you to add Lightning Pages to Lightning Experience and the mobile app." A sidebar on the left shows 'User Interface' sections for 'Rename Tabs and Labels' and 'Tabs'.

- Email alerts

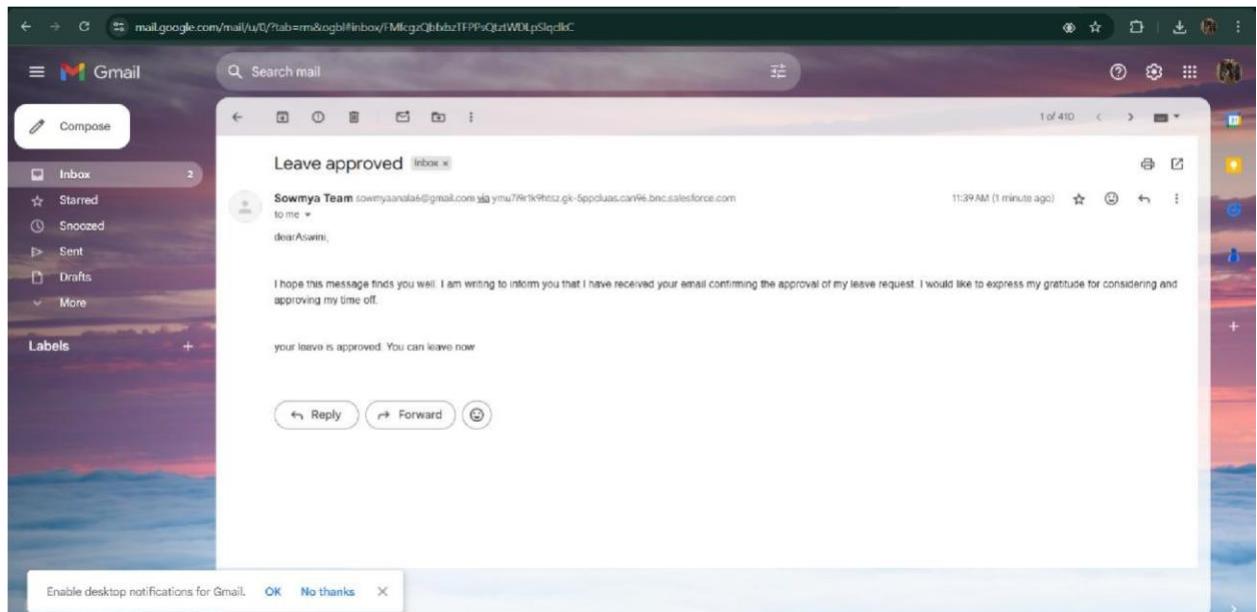
The screenshot shows the 'Lease Management' system interface. The top navigation bar includes 'Lease Management', 'Payment', 'Tenants', 'property', and 'lease'. Below the navigation is a 'Tenants > nianjian Approval History' section. A table titled 'Approval History' lists 8 items, sorted by Is Pending, updated a few seconds ago. The columns include Step Name, Date, Status, Assigned To, Actual Approver, and Comments. The data in the table 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

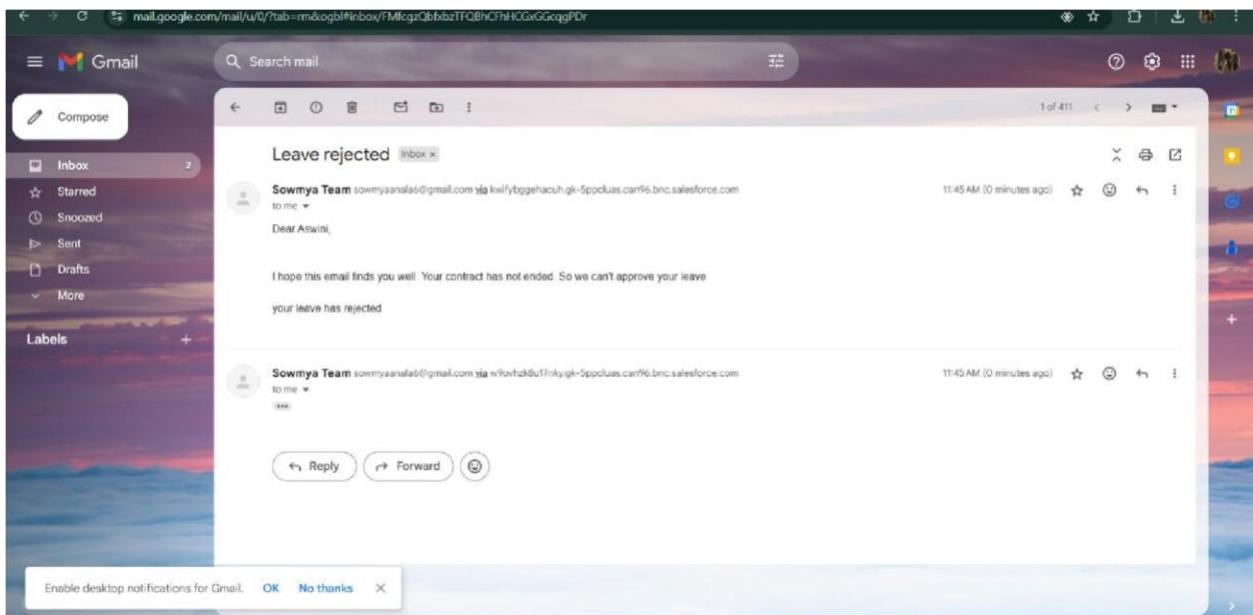
- Request for approve the leave



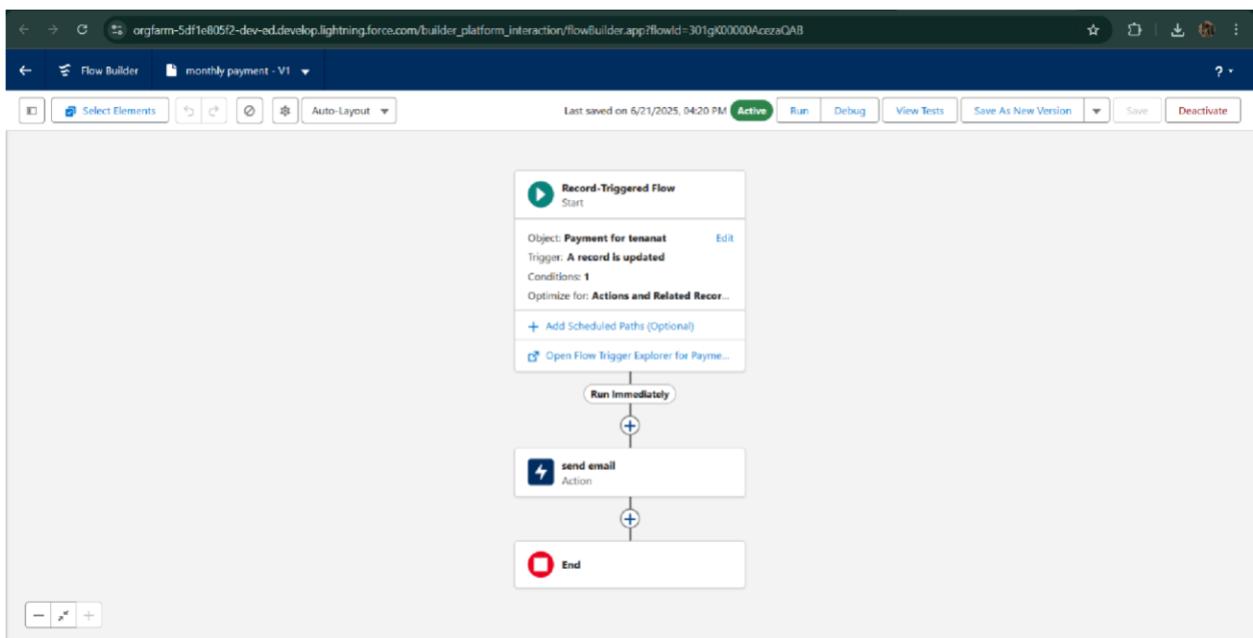
- Leave approved



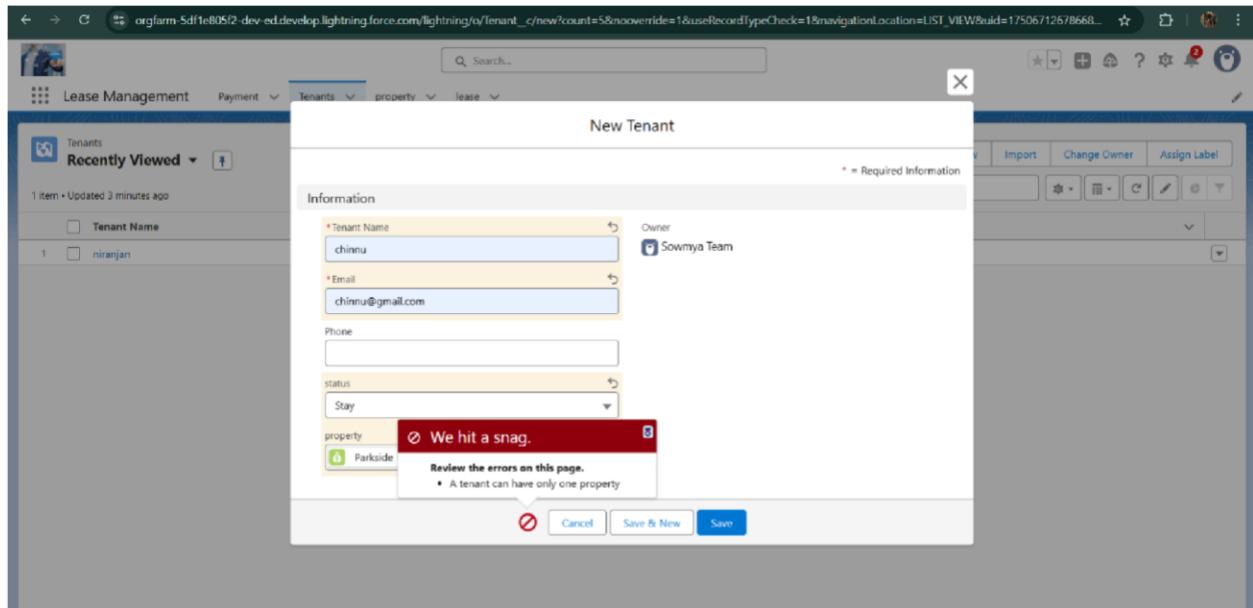
- Leave rejected



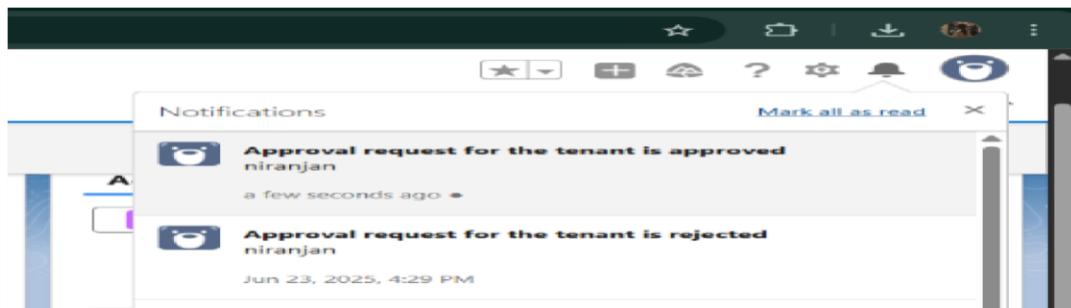
- Flow runs



- Trigger error messages



- Approval process notifications



ADVANTAGES & DISADVANTAGES

CONCLUSION

The Lease Management System successfully streamlines the operations of leasing through a structured, automated Salesforce application. It improves efficiency, communication, and data accuracy for both admins and tenants.

APPENDIX

- **Source Code:** Provided in Apex Classes and Triggers

Test.apxt: trigger test on Tenant__c

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

} } **testHandler.apxc:**

```
public class
```

```
testHandler { public
```

```
static void
```

```
preventInsert(List<
```

```
Tenant__c> newlist)
```

```
{ Set<Id>
```

```
existingPropertyIds
```

```
= new Set<Id>()
```

```
for (Tenant__c existingTenant : [SELECT Id, Property__c FROM Tenant__c
WHERE Property__c != null]) {
```

```
existingPropertyIds.add(existingTenant.Property__c;
```

```
} for (Tenant__c newTenant :
```

```

newlist) {

    if (newTenant.Property__c != null &&
existingPropertyIds.contains(newTenant.Property__c)) { newTenant.addError('A

tenant can have only one property');

}

}

}

}

```

MothlyEmailScheduler.apxc:

```

global class MonthlyEmailScheduler implements Schedulable {
    global

    void execute(SchedulableContext sc) {
        Integer currentDay =
Date.today().day();
        if (currentDay == 1) {

            sendMonthlyEmails();
        }
    }

    public static void
sendMonthlyEmails() {
    List<Tenant__c>
tenants = [SELECT Id, Email__c FROM
Tenant__c];
    for (Tenant__c tenant :
tenants) {

        String recipientEmail = tenant.Email__c;
        String emailContent =
'I trust this email finds you well. I am writing to remind you
that the monthly rent is due. Your timely payment ensures the smooth functioning of our
rental arrangement and helps maintain a positive living environment for all.';

        String emailSubject = 'Reminder: Monthly Rent Payment Due';
        Messaging.SingleEmailMessage email = new

```

```
        Messaging.SingleEmailMessage(); email.setToAddresses(new
String[]{recipientEmail}); email.setSubject(emailSubject);
email.setPlainTextBody(emailContent);

        Messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});

    }

}
```