

LEASE MANAGEMENT

College Name: A.V.P College of Arts and Science
College Code: bruaj

TEAM ID: NM2025PMIB25334

TEAM MEMBERS:

Team Leader Name: Gayathri.S

Email id: gayathritamilselvi0305@gmail.com

Team Member 1: Asma.S

Email: asmasyedismail7@gmail.com

Team Member 2: Moulisha.S

Email: moulishaselvakumar@gmail.com

Team Member 2: Priyanka.S

Email: priyankasundarraj37@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.



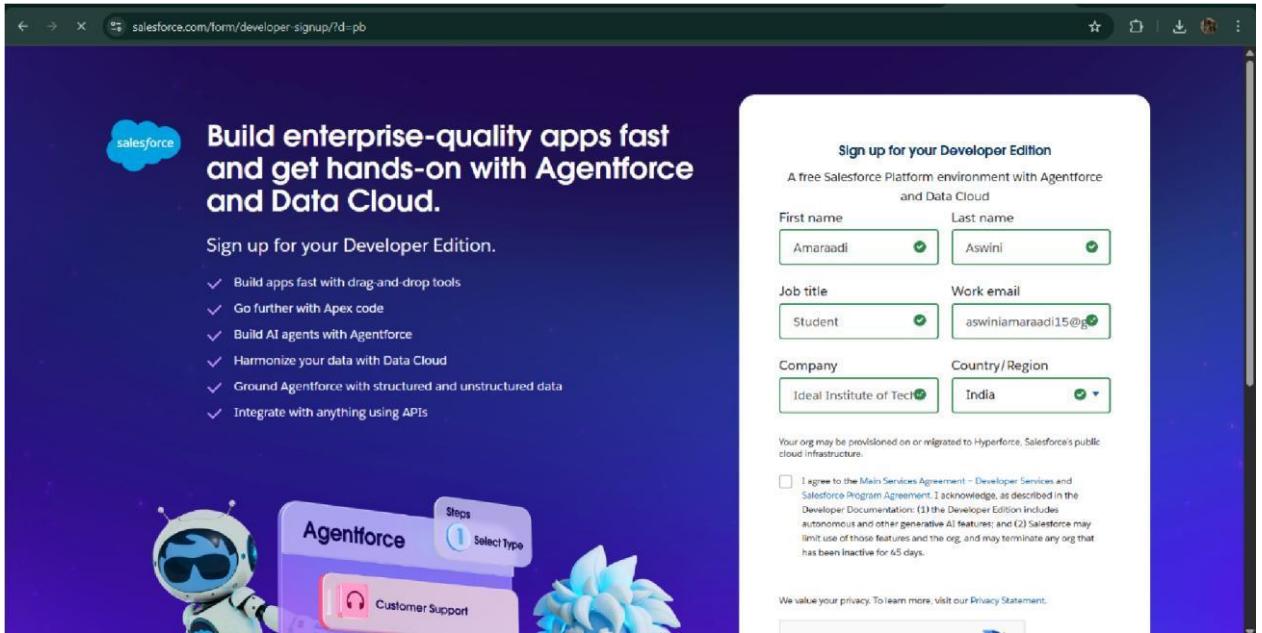
1.2 Purpose

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

The screenshot shows the Salesforce Object Manager interface. The URL in the address bar is <https://orgname-5df1e80512-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgk000000JTT/Details/view>. The page title is "SETUP > OBJECT MANAGER property". The left sidebar lists various object settings like 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 Scoring Rules. The main content area displays the "Details" tab for the "property" object. The "Details" section includes fields for API Name (property__c), Singular Label (property), Plural Label (property), and various checkboxes for Enable Reports, Track Activities, Track Field History, Deployment Status (Deployed), Help Settings, and Standard salesforce.com Help Window. There are "Edit" and "Delete" buttons at the top right of the details section.

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

Edit Delete

This screenshot shows the Salesforce Object Manager interface for the 'Tenant' object. The left sidebar lists various configuration options like Fields & Relationships, Page Layouts, and Record Types. The main 'Details' tab is selected, showing the API name 'Tenant__c', a custom label 'Tenant', and a plural label 'Tenants'. Under 'Enable Reports', 'Track Activities' and 'Track Field History' are checked. Deployment status is set to 'Deployed'. Help settings point to the standard Salesforce help window.

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

Edit Delete

This screenshot shows the Salesforce Object Manager interface for the 'lease' object. Similar to the 'Tenant' object, it has an API name 'lease__c', a custom label 'lease', and a plural label 'lease'. It also includes 'Enable Reports' settings for tracking activities and field history, and is currently deployed. The configuration options listed on the left are identical to the 'Tenant' object's.

Payment for tenant

Details

Description

API Name: `Payment_for_tenant_c`

Custom

Singular Label: **Payment for tenant**

Plural Label: **Payments**

Enable Reports: ✓

Track Activities: ✓

Track Field History: ✓

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

- Configured fields and relationships

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)	✓	
sfqt	sfqt__c	Text(18)		
Type	Type__c	Picklist		

Setup > Object Manager

Payment for tenant

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

Setup > Object Manager

lease

Fields & Relationships				
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
End date	End_date_c	Date		
Last Modified By	LastModifiedById	Lookup(User)		
lease Name	Name	Text(80)		✓
Owner	OwnerId	Lookup(User, Group)		✓
property	property_c	Lookup(property)		✓
start date	start_date_c	Date		

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: Placeholder image

Primary Color Hex Value: #0070D2

Org Theme Options: Use the app's image and color instead of the org's custom theme

App Launcher Preview

Lease Management
Application to efficiently handle the processes relate...

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

App Settings

Navigation Items

Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.

Available Items

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

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

Selected Items

- Payment
- Tenants
- property
- lease

Up ▲ Down ▼

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

App Settings

User Profiles

Choose the user profiles that can access this app.

Available Profiles

Type to filter list...

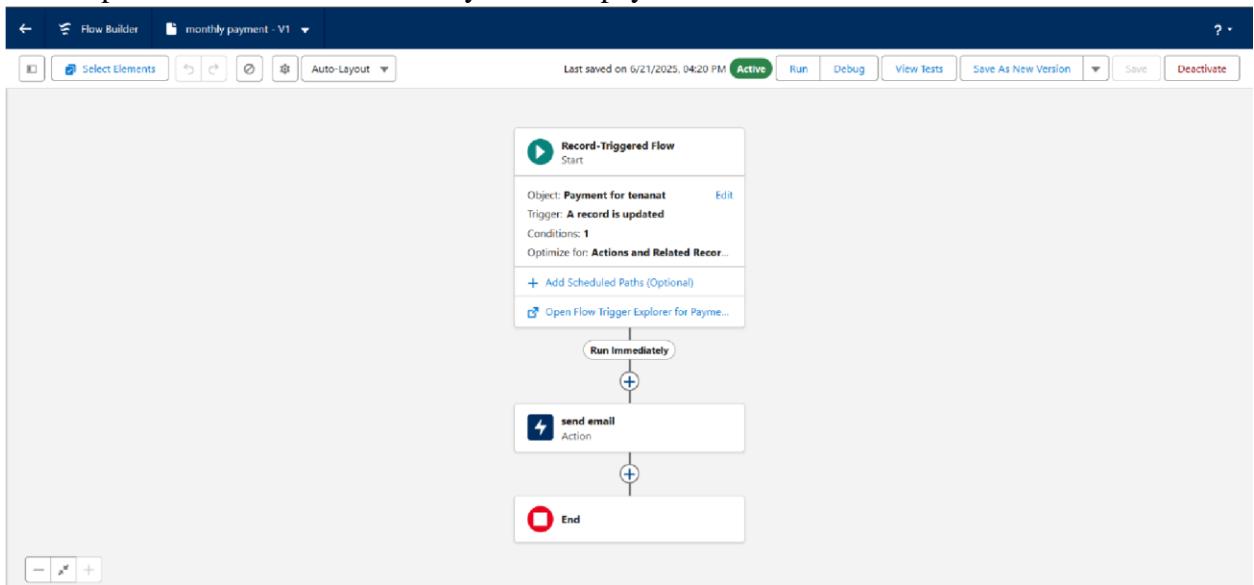
- Analytics Cloud Integration User
- Analytics Cloud Security User
- Anypoint Integration
- Authenticated Website
- Authenticated Website
- B2B Reordering Portal Buyer Profile
- Contract Manager
- Custom: Marketing Profile
- Custom: Sales Profile
- Custom: Support Profile
- Customer Community Login User

Selected Profiles

- System Administrator

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

- Implemented Flows for monthly rent and payment success

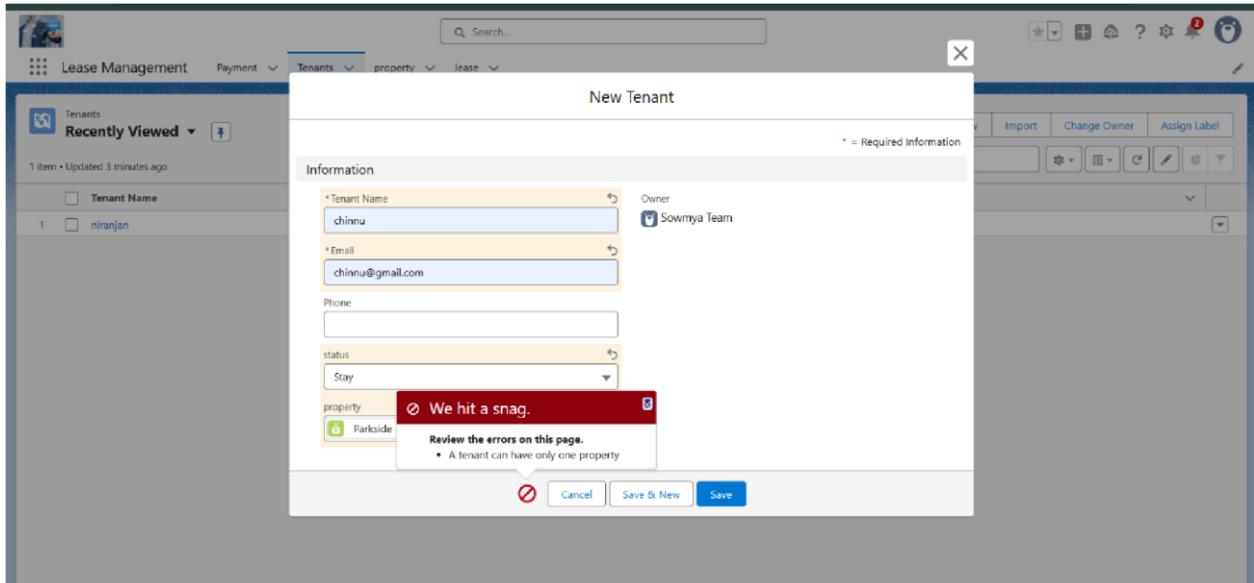


- To create a validation rule to a Lease Object

The screenshot shows the 'Validation Rule Edit' screen for the 'lease' object. The 'Rule Name' is 'lease_end_date'. The 'Error Condition Formula' is set to '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 'Active' checkbox is checked.

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

- Added Apex trigger to restrict multiple tenants per property



- Scheduled monthly reminder emails using Apex class

```

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

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

The screenshot shows the Salesforce Setup interface with the search bar set to 'email template'. Under the 'Email' section, 'Classic Email Templates' is selected. A search result for 'Leave approved' is displayed. The 'Email Template Detail' section shows the following information:

- Email Template Name:** Leave approved
- Template Unique Name:** Leave_approved
- Encoding:** Unicode (UTF-8)
- Author:** Sowmya Team [Changes]
- Description:** Created By Sowmya Team, 6/20/2025, 1:06 AM
- Modified By:** Sowmya Team, 6/20/2025, 1:06 AM
- Available For Use:** checked
- Last Used Date:** Not specified
- Times Used:** Not specified

The 'Email Template' preview area shows the following content:

```

Subject: Leave approved
Main Text Preview:
dear{Tenant__c.Name},

I hope this message finds you well. I am writing to inform you that I have received your email confirming the approval of my leave request. I would like to express my gratitude for considering and approving my time off.

your leave is approved. You can leave now.
  
```

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

- Email Template Name:** tenant leaving
- Template Unique Name:** tenant_leaving
- Encoding:** Unicode (UTF-8)
- Author:** Sowmya Team [Changes]
- Description:** Created By Sowmya Team, 6/20/2025, 1:06 AM
- Modified By:** Sowmya Team, 6/20/2025, 1:06 AM
- Available For Use:** checked
- Last Used Date:** Not specified
- Times Used:** Not specified

The 'Email Template' preview area shows the following content:

```

Subject: request for approve the leave
Main 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

Field	Value
Email Templates from Salesforce	Unfiled Public Classic Email Templates
Email Template Name	Leave rejected
Template Unique Name	Leave_rejected
Encoding	Unicode (UTF-8)
Author	Sowmya Team [Change]
Description	
Created By	Sowmya Team 6/20/2025, 1:11 AM
Modified By	Sowmya Team 6/20/2025, 1:11 AM

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

Field	Value
Email Templates from Salesforce	Unfiled 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

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.

Email Template Detail

- 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

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:

Process Definition Detail

- Process Name: TenantApproval
- Unique Name: TenantApproval
- Description: Tenant: status equals Stay
- Entry Criteria: Record Editability: Administrator ONLY
- 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:

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

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

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

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

● Apex Trigger

Create an Apex Trigger

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

- trigger test on Tenant_c (before insert)** is the Apex trigger code.
- A modal window titled "Open" is displayed, listing various entity types:

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

Developer Console - Google Chrome

File Edit Debug Test Workspace Help < >

test.apex [testHandler.apc] MonthlyEmailScheduler.apc []

Code Coverage Name: API Version: 64 Go To

```

1 trigger test on Tenant__c (before insert)
2
3 {
4
5     if(trigger.isInsert && trigger.isBefore){
6
7         testHandler.preventInsert(trigger.new);
8
9     }
10}
11

```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

Create an Apex Handler class

Developer Console - Google Chrome

File Edit Debug Test Workspace Help < >

test.apex [testHandler.apc] MonthlyEmailScheduler.apc []

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(exi
10
11         }
12
13
14         for (Tenant__c newTenant : newList) {
15
16
17             if (newTenant.Property__c != null) {
18
19                 newTenantaddError('A
20
21             }
22
23     }

```

Entity Type Entity Name Namespace Related

Entity Type	Entity Name	Namespace	Related
Classes	testHandler	MonthlyEmailScheduler	← test ApexTrigger ← property CustomField ← Tenant__c Object ← Tenant__c Object
Triggers			
Pages			
Page Components			
Objects			
Static Resources			
Packages			

Open Filter Hide Managed Packages Refresh

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

Developer Console - Google Chrome

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

File Edit Debug Test Workspace Help

test.apex testHandler.apxc MonthlyEmailScheduler.apac

Code Coverage Name API Version 64 Go To

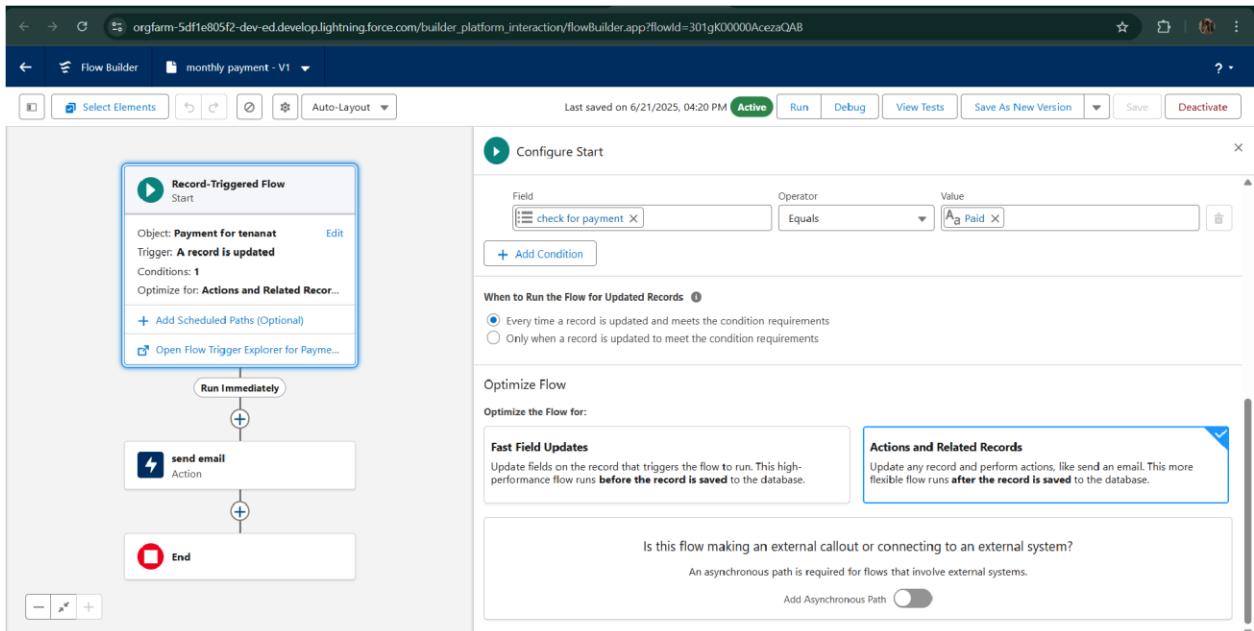
```

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         }
24
25     }
26
27 }
```

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

● FLOWS



Record-Triggered Flow Start

Object: **Payment for tenant** Edit
Trigger: **A record is updated**
Conditions: 1
Optimize for: **Actions and Related Records**

+ Add Scheduled Paths (Optional)

Open Flow Trigger Explorer for Payment for tenant...

Run Immediately

send email Action

End

Configure Start

Select Object

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.

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)

- Schedule class:
Create an Apex Class

```

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

```

Entity Type

Entity Type	Entities	Related
Classes	testHandler	Name Extent CrossTrigger References
Triggers	MonthlyEmailScheduler	Email CustomField CObject References
Pages		Tenant__c SOObject References
Page Components		
Objects		
Static Resources		
Packages		

```

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 = 'Rent Remind: Monthly Rent Payment Due';
27
28            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
29
30            email.setToNames(recipientEmail);
31            email.setSubject(emailSubject);
32            email.setPlainTextBody(emailContent);
33
34            messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
35
36        }
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 with 'apex' typed in. The main area displays the 'Apex Classes' section for the 'Apex Class' named 'MonthlyEmailScheduler'. The 'Apex Class Detail' table shows the following information:

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

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
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 = 'Rent Remind: Monthly Rent Payment Due';
27
28            Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
29
30            email.setToNames(recipientEmail);
31            email.setSubject(emailSubject);
32            email.setPlainTextBody(emailContent);
33
34            messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
35
36        }
37
38    }
39
40 }
41
42

```

Screenshot of the Salesforce Lightning interface showing the 'Lease Management' tab selected. A context menu is open over a tenant record named 'Aswini'. The menu options include:

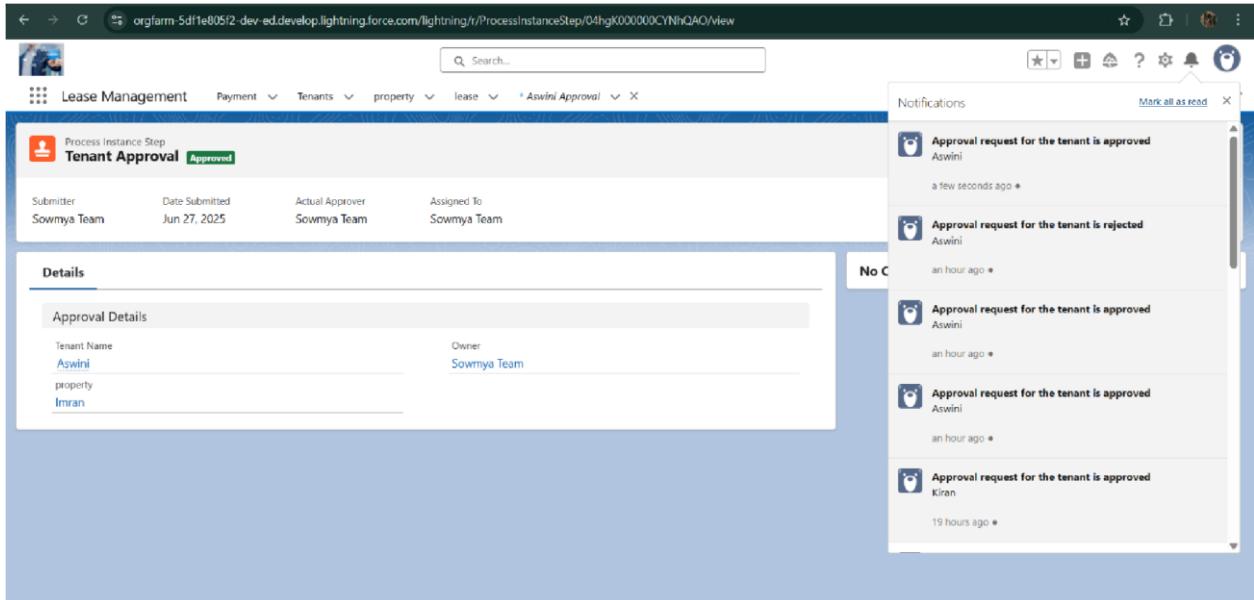
- New Case
- New Lead
- Delete
- Clone
- Change Owner
- Printable View
- Submit for Approval**
- Edit Labels

The 'Submit for Approval' option is highlighted.

Screenshot of the Salesforce Lightning interface showing the 'Lease Management' tab selected. A success message 'Tenant was submitted for approval.' is displayed in a green banner at the top right. The tenant record for 'Aswini' is shown with all fields filled in, including:

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

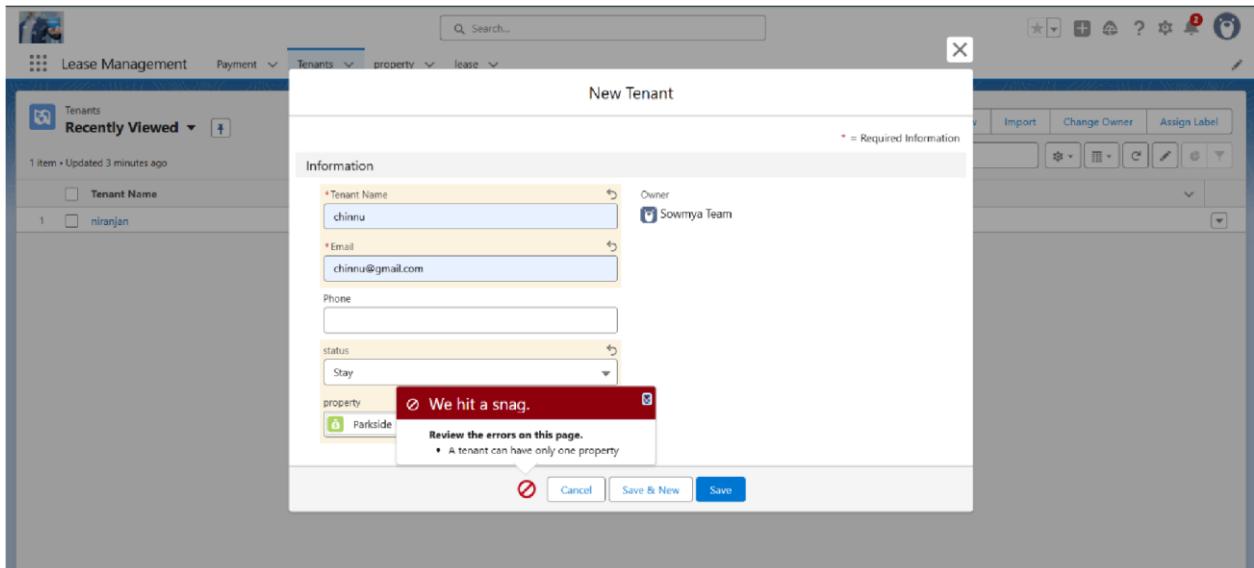
The 'Save' button is visible at the bottom right.



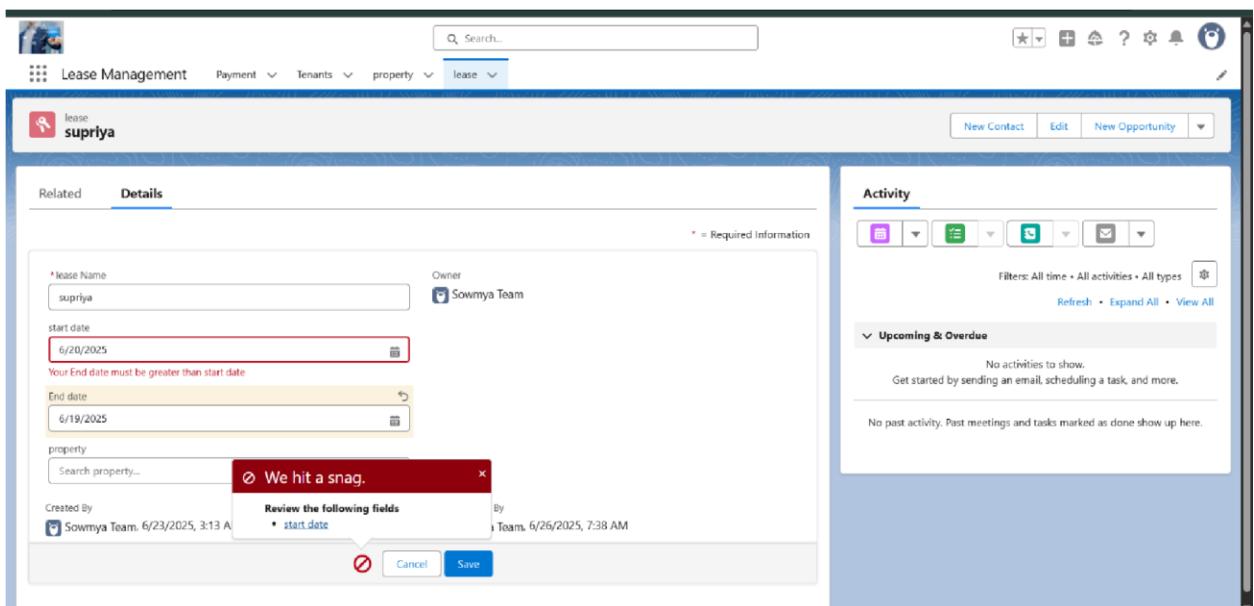
FUNCTIONAL AND PERFORMANCE TESTING

Performance Testing

- Trigger validation by entering duplicate tenant-property records

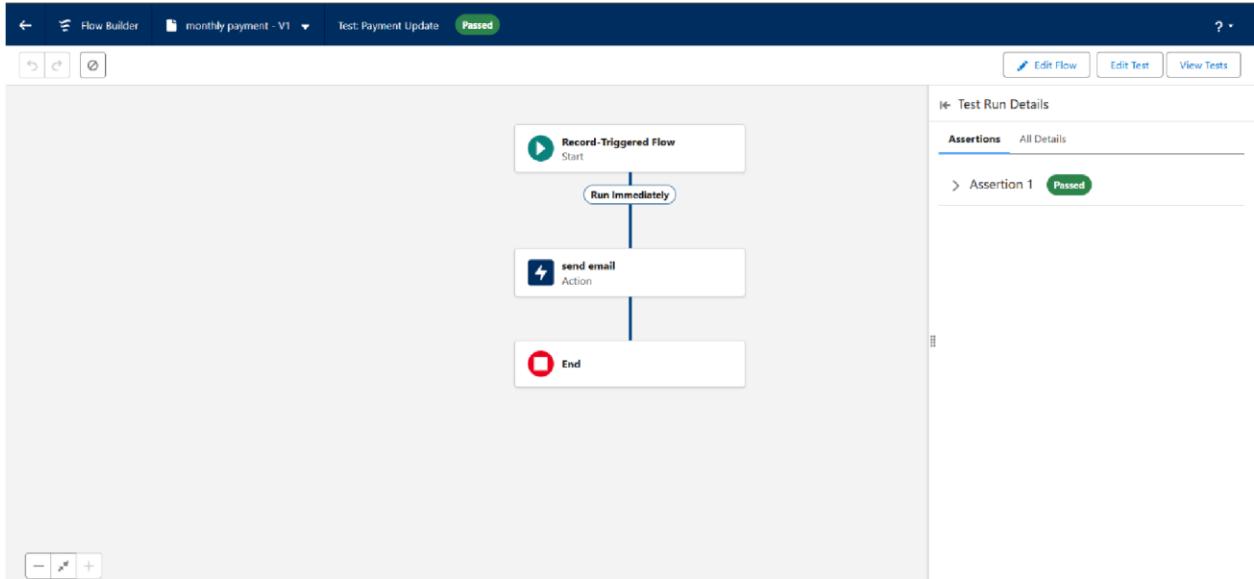


- Validation Rule checking



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

- Test flows on payment update



- Approval process validated through email alerts and status updates

The screenshot shows a CRM interface for 'Lease Management'. On the left, a 'Tenant' record for 'niranjan' is displayed under the 'Tenants' tab. The 'Details' tab is selected, showing fields like Tenant Name (niranjan), Email (niranjan1506@gmail.com), Owner (Sowmya Team), Phone, Status (Stay), and Property (Parkside Lofts). It also shows creation and modification details by the Sowmya Team. On the right, a 'Notifications' sidebar is open, listing several recent notifications: 'Approval request for the tenant is approved' (niranjan, a few seconds ago), 'Approval request for the tenant is rejected' (niranjan, Jun 23, 2025, 4:29 PM), 'Approval request for the tenant is approved' (niranjan, Jun 23, 2025, 4:25 PM), 'Approval request for the tenant is approved' (niranjan, Jun 23, 2025, 4:14 PM), and a 'New Guidance Center learning resource available' (Jun 20, 2025, 1:28 PM).

The screenshot shows a CRM interface for 'Lease Management'. On the left, the 'Approval History' section displays a list of steps: Step 1 (Approved, 6/25/2025, 5:39 AM), Approval Request Submitted (Submitted, 6/25/2025, 5:39 AM), Step 1 (Rejected, 6/23/2025, 3:59 AM), Approval Request Submitted (Submitted, 6/23/2025, 3:58 AM), Step 1 (Approved, 6/23/2025, 3:55 AM), and Approval Request Submitted (Submitted, 6/23/2025, 3:55 AM). Below this is a 'View All' button. On the right, the 'Payment' section shows two entries: 'Jack' and 'Rahul', with a 'View All' button. A sidebar on the right indicates 'No past activity. Past meetings and tasks marked as done show up here.'

RESULTS

Output Screenshots

- Tabs for Property, Tenant, Lease, Payment

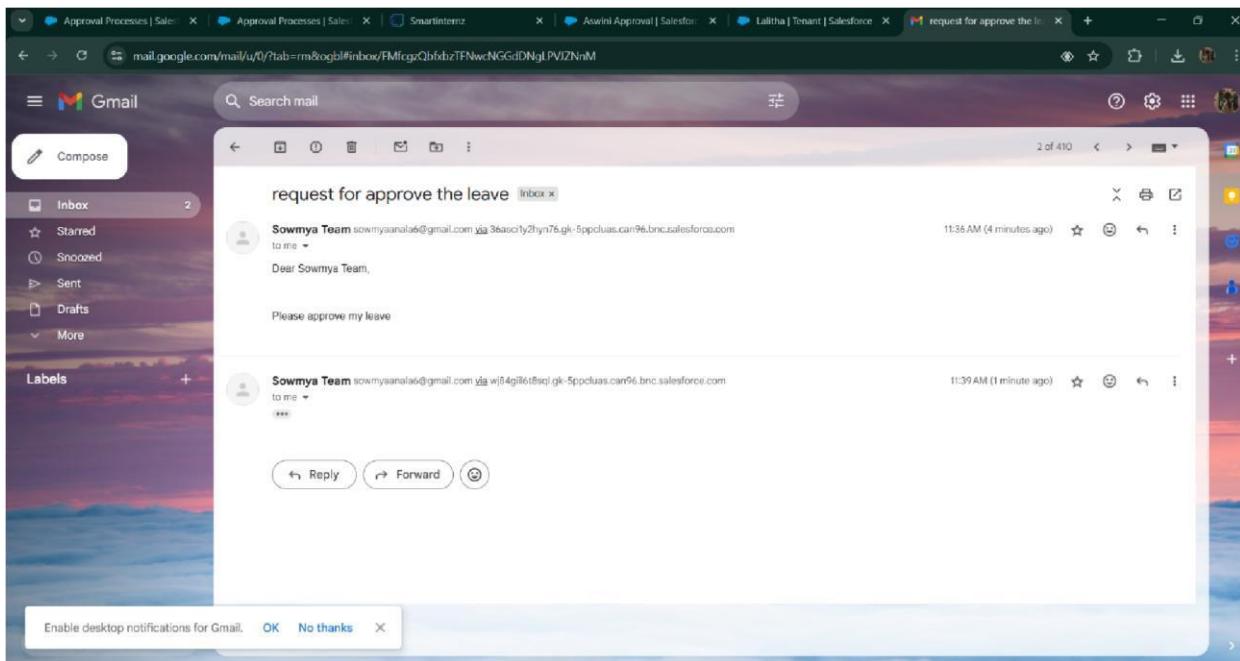
The screenshot shows the Salesforce Setup interface under the 'Tabs' section. It displays four custom tabs: 'Lease' (Tab Style: Keys), 'Payment' (Tab Style: Credit card), 'property' (Tab Style: Sack), and 'Tenant' (Tab Style: Map). The 'Custom Object Tabs' section also lists these tabs. Below these are sections for 'Web Tabs' and 'Visualforce Tabs', both of which are currently empty.

- Email alerts

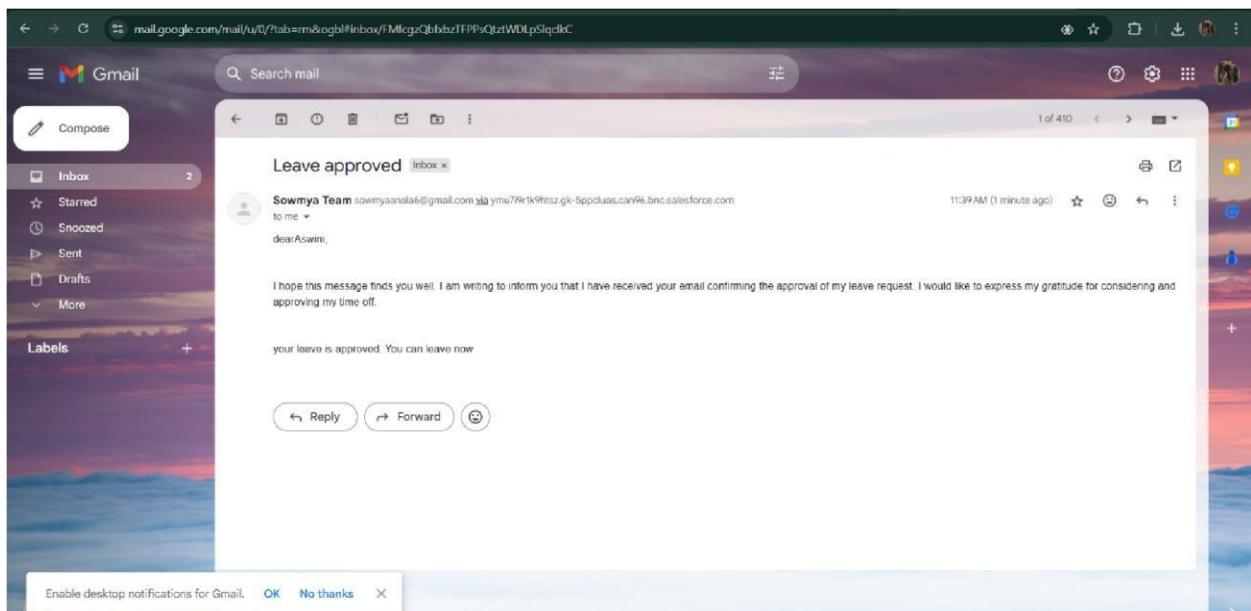
The screenshot shows the 'Approval History' section of the 'Lease Management' application. It lists eight approval steps for a tenant named 'niranjana'. The steps are: Step 1 (Approved, Assigned To: Sowmya Team, Comments: approved), Approval Request Submitted (Submitted, Assigned To: Sowmya Team, Comments: leaving), Step 1 (Rejected, Assigned To: Sowmya Team, Comments: Rejected), Approval Request Submitted (Submitted, Assigned To: Sowmya Team, Comments: Leaving), Step 1 (Approved, Assigned To: Sowmya Team, Comments: Approved), Approval Request Submitted (Submitted, Assigned To: Sowmya Team, Comments: leaving), Step 1 (Approved, Assigned To: Sowmya Team, Comments: Approval Approved), and Approval Request Submitted (Submitted, Assigned To: Sowmya Team, Comments: Leaving).

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

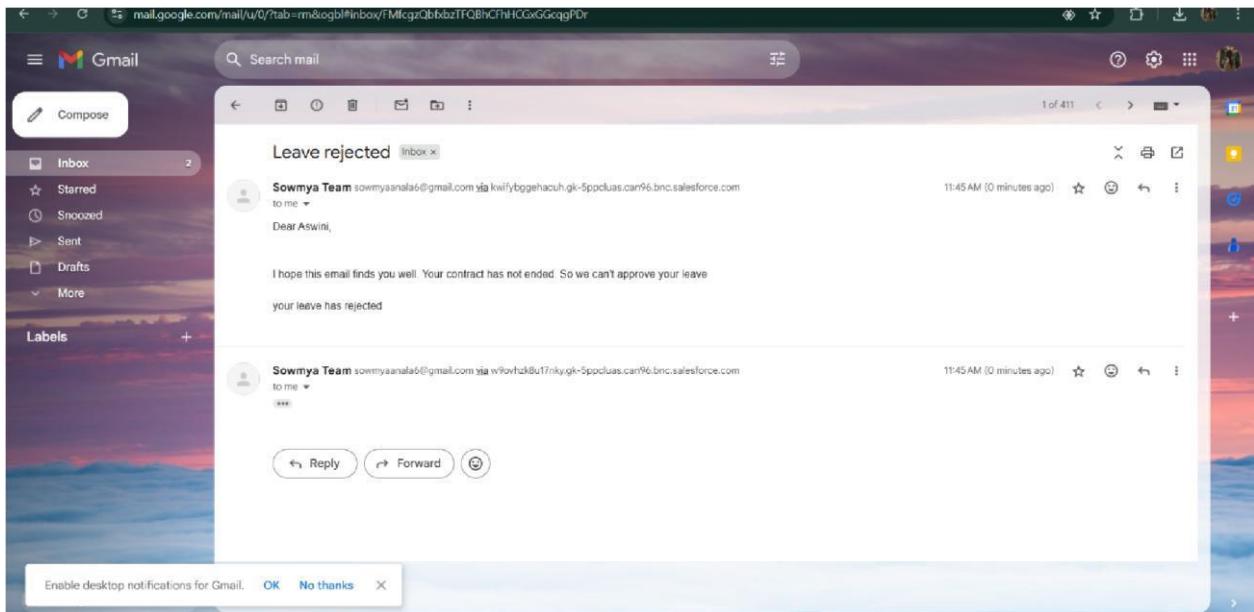
- Request for approve the leave



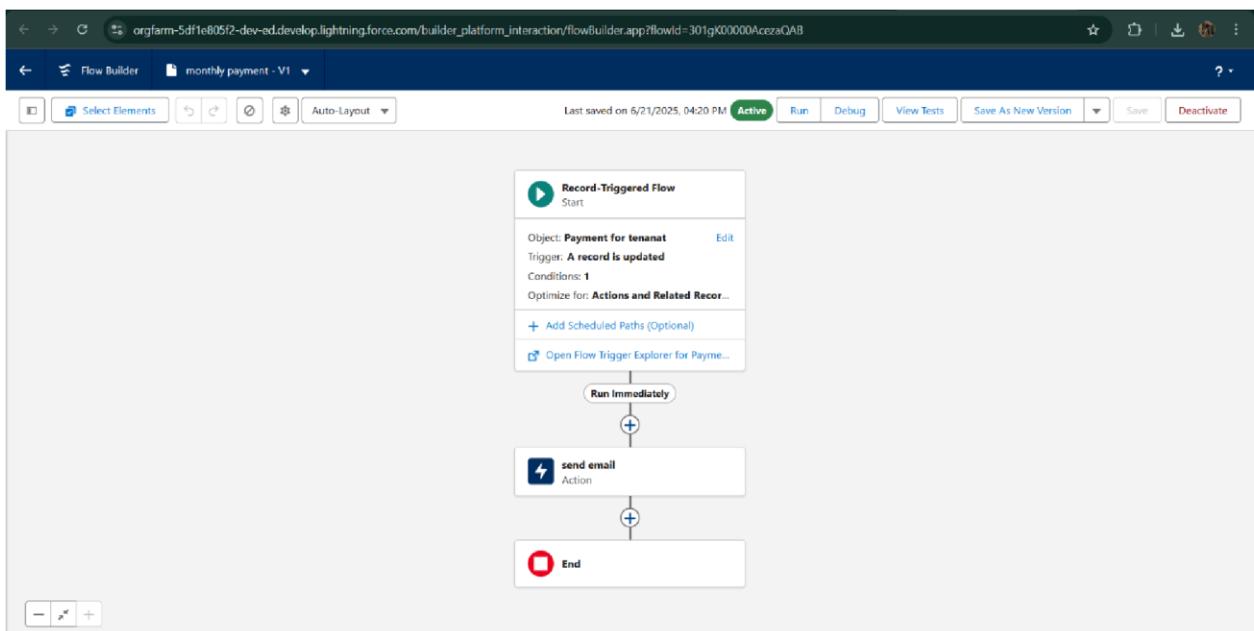
- Leave approved



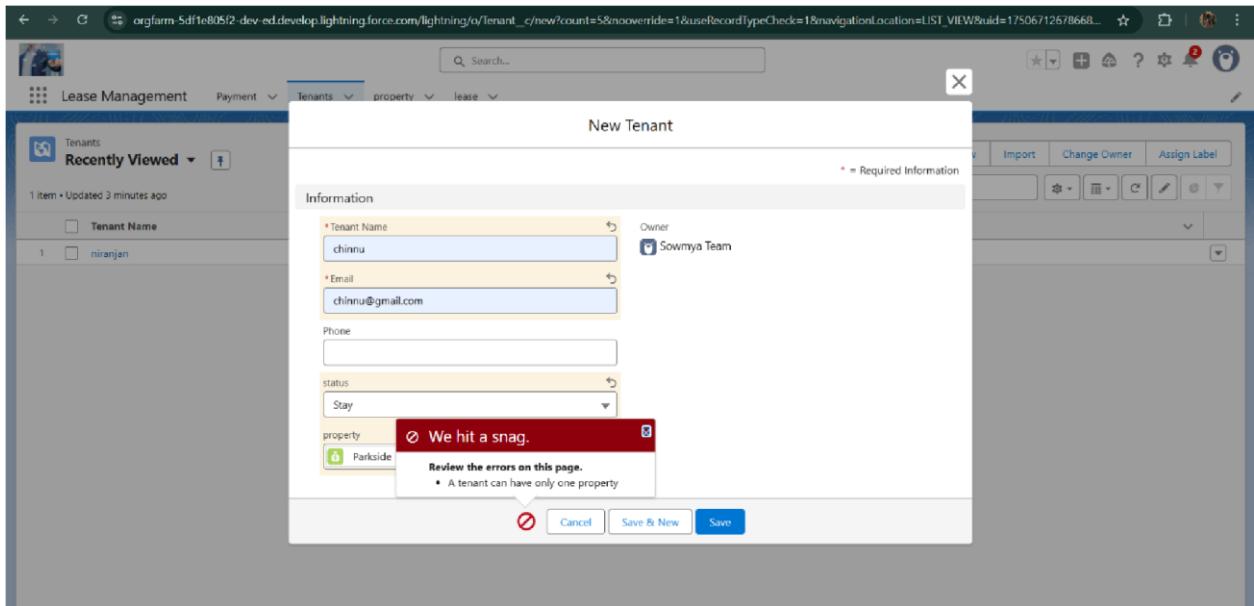
- Leave rejected



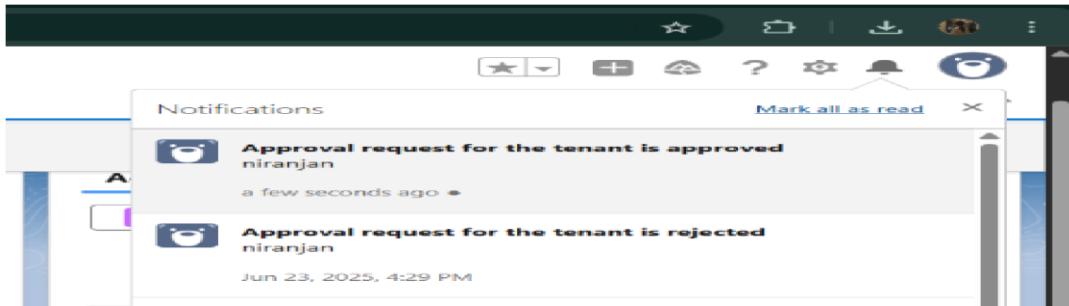
- Flow runs



- 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});  
    }  
}  
}
```