

# PROJECT TITLE

College Name: *L.R.G Arts College, Tirupur*

College Code: *bru07*

**TEAM ID:** *NM2025TMID28014*

## **TEAM MEMBERS:**

**Team LeaderName:** *Priyavarshinisree.S*

**Email:** *santhakumarpriya47@gmail.com*

**Team Member1:** *Divya.M*

**Email:** *divyamsv2006@gmail.com*

**Team Member:** *Karthiyayini.S*

**Email:** *sureshmohana45@gmail.com*

**Team Member:** *Vanmathi.D*

**Email:** *vanmathiv902@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.



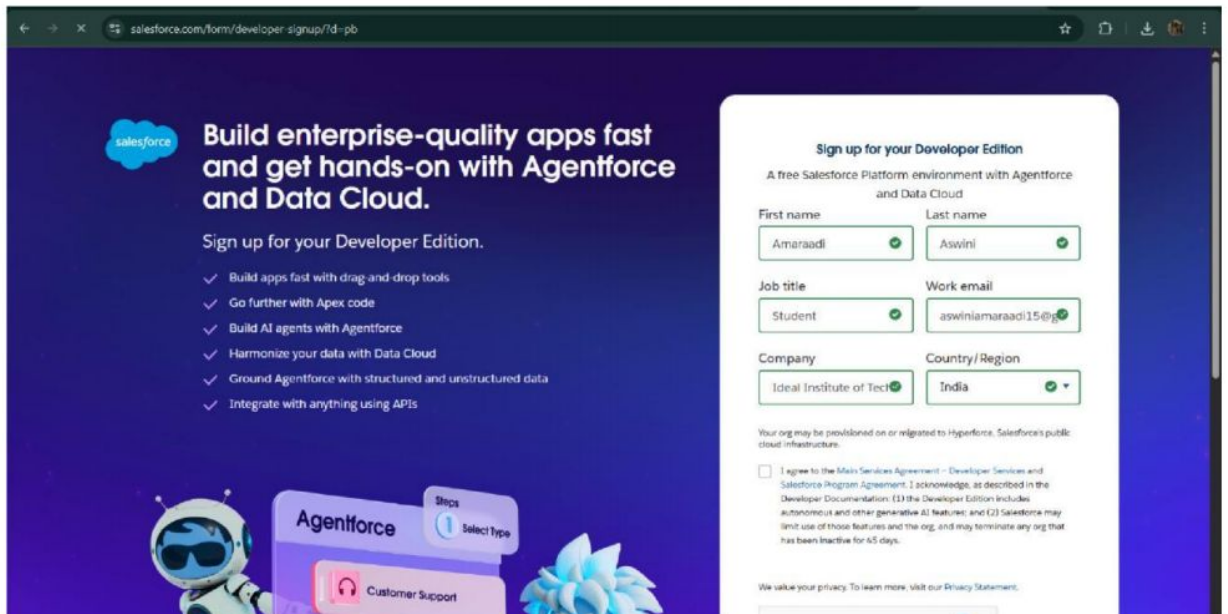
## 1.2 Purpose

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>



salesforce.com/form/developer-signup/7d-pb

**Build enterprise-quality apps fast and get hands-on with Agentforce and Data Cloud.**

Sign up for your Developer Edition.

- ✓ Build apps fast with drag-and-drop tools
- ✓ Go further with Apex code
- ✓ Build AI agents with Agentforce
- ✓ Harmonize your data with Data Cloud
- ✓ Ground Agentforce with structured and unstructured data
- ✓ Integrate with anything using APIs

**Sign up for your Developer Edition**  
A free Salesforce Platform environment with Agentforce and Data Cloud

First name: Amaraadi ✓ Last name: Aswini ✓

Job title: Student ✓ Work email: aswiniamaraadi15@gmail.com ✓

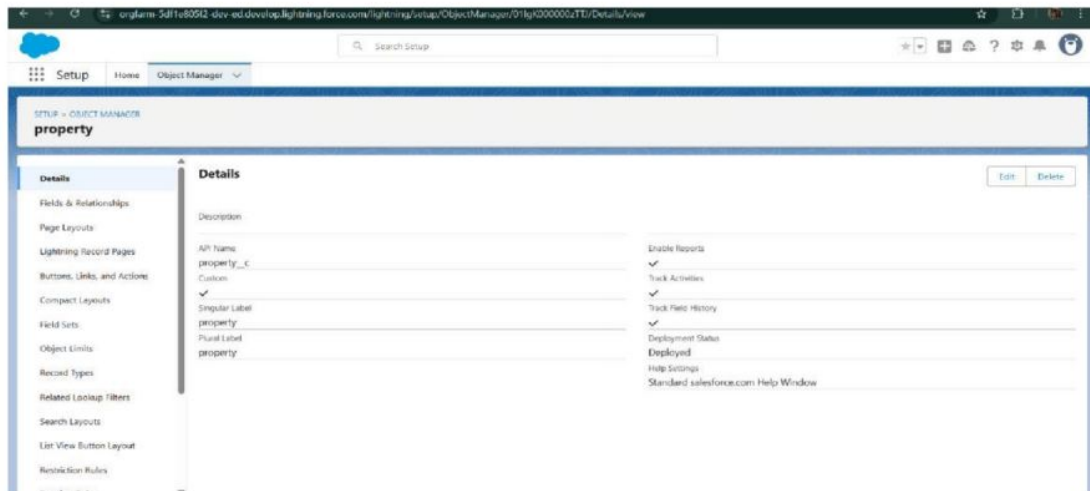
Company: Ideal Institute of Tech ✓ Country/Region: India ✓

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

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

We value your privacy. To learn more, visit our [Privacy Statement](#).

- Created objects: Property, Tenant, Lease, Payment



Setup - Object Manager - Details

**property**

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

Automation Rules

Details

Description

API Name: property\_\_c

Custom

✓ Singular Label: property

Plural Label: property

Enable Reports: ✓

Track Activities: ✓

Track Field History: ✓

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

Edit Delete

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

Search Setup

Setup Home Object Manager

SETUP > OBJECT MANAGER

### Tenant

**Details**

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

**Details**

Edit Delete

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

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

Search Setup

Setup Home Object Manager

SETUP > OBJECT MANAGER

### lease

**Details**

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

**Details**

Edit Delete

Description

API Name  
lease\_\_c

Custom

Singular Label  
lease

Plural Label  
leases

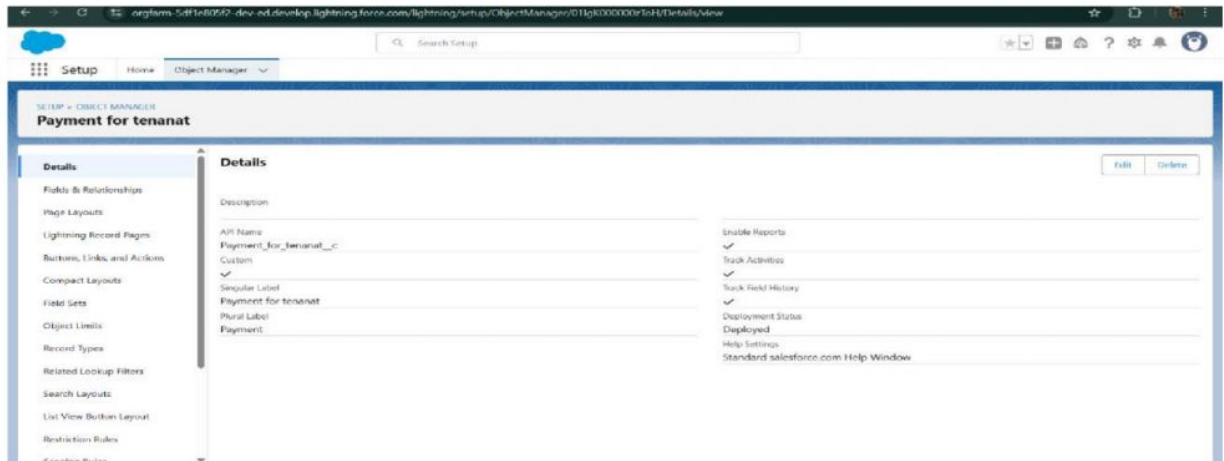
Enable Reports  
✓

Track Activities  
✓

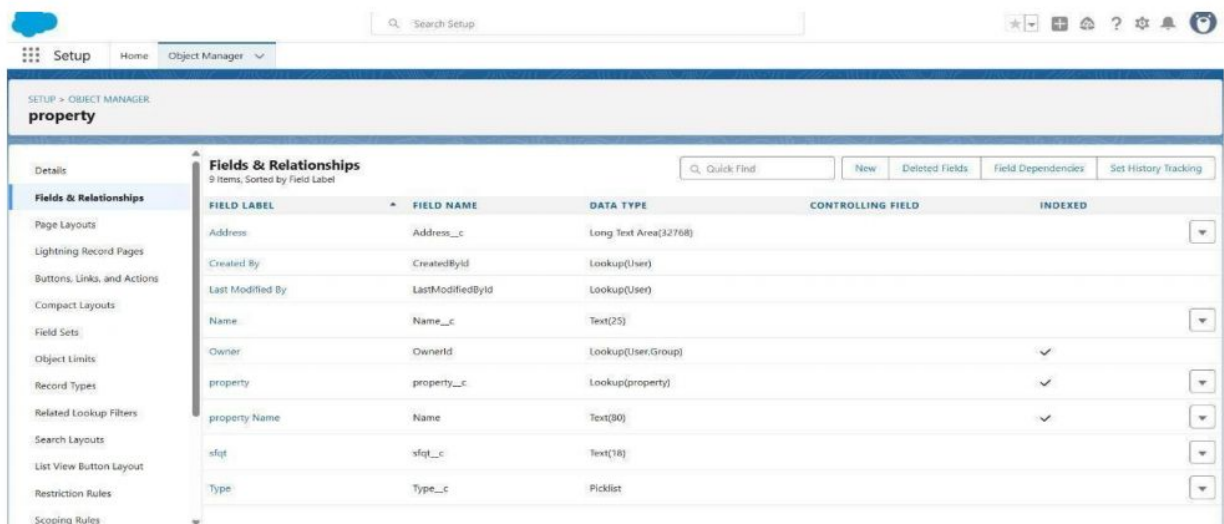
Track Field History  
✓

Deployment Status  
Deployed

Help Settings  
Standard salesforce.com Help Window



- Configured fields and relationships



Setup

Home

Object Manager

Search Setup

Fields & Relationships

7 Items, Sorted by Field Label

Quick Find

New

Deleted Fields

Field Dependencies

Set History Tracking

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

Home

Object Manager

Search Setup

lease

Fields & Relationships

7 Items, Sorted by Field Label

Quick Find

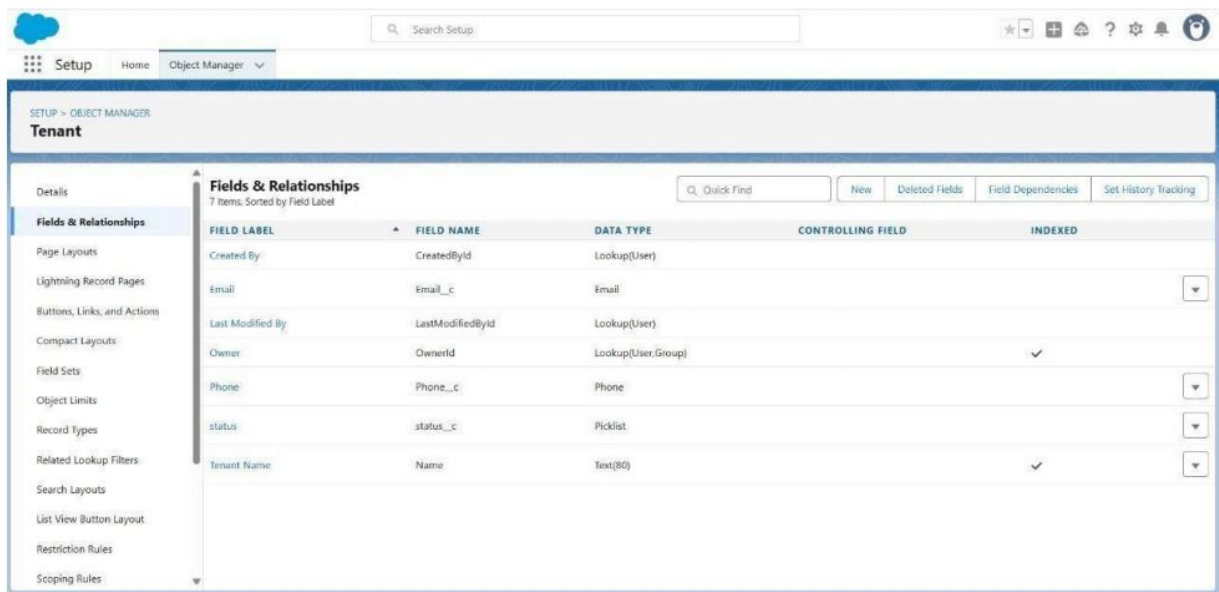
New

Deleted Fields

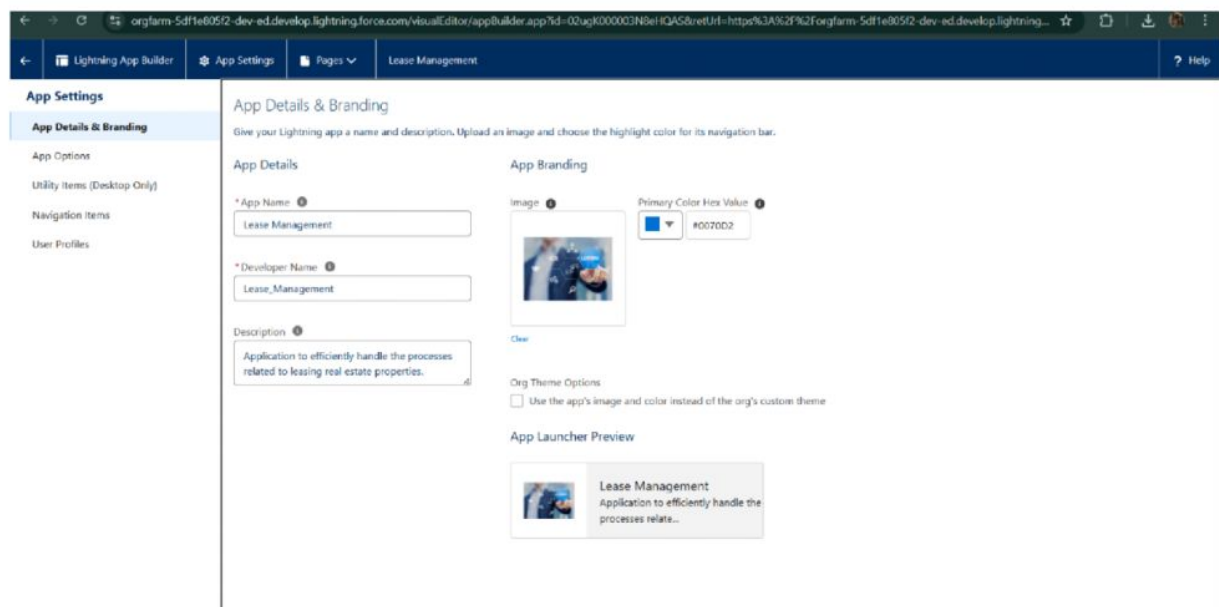
Field Dependencies

Set History Tracking

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		



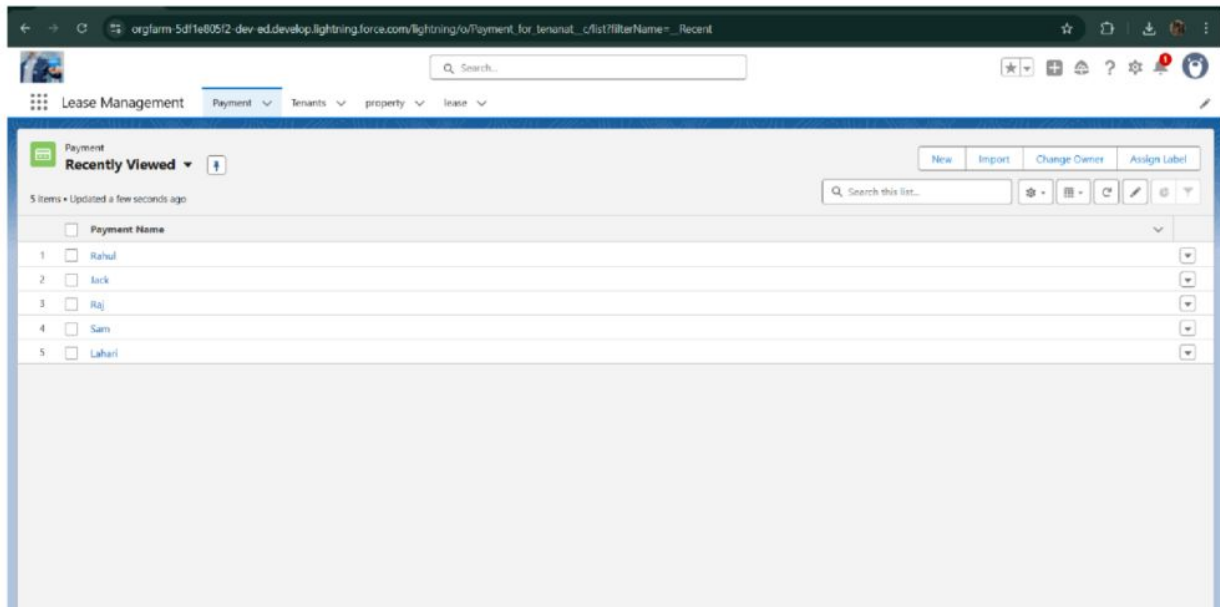
- Developed Lightning App with relevant tabs



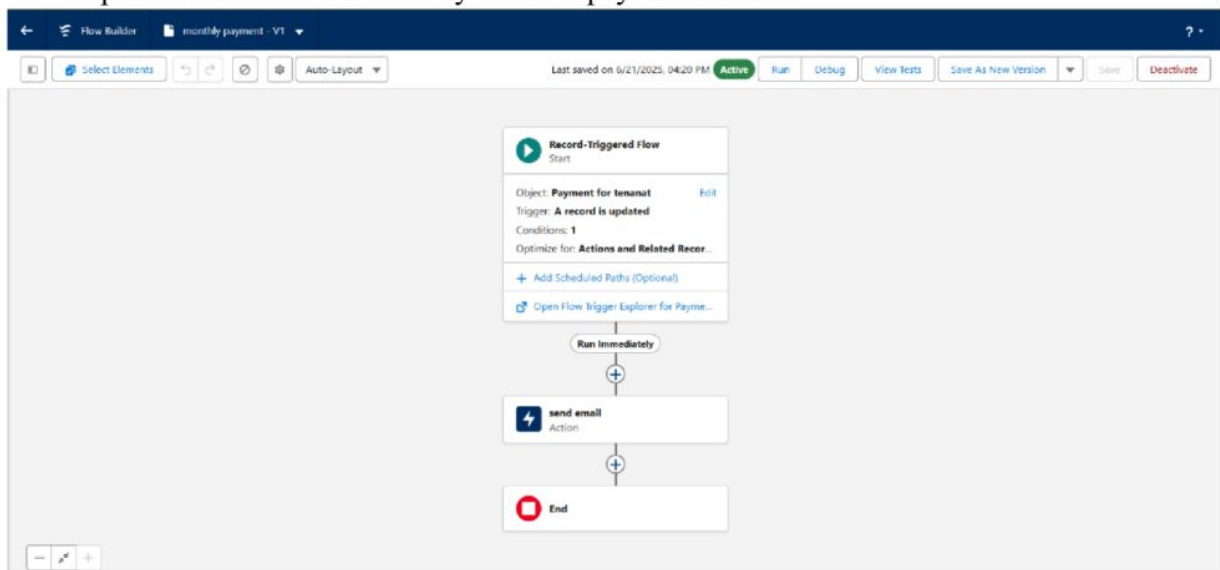




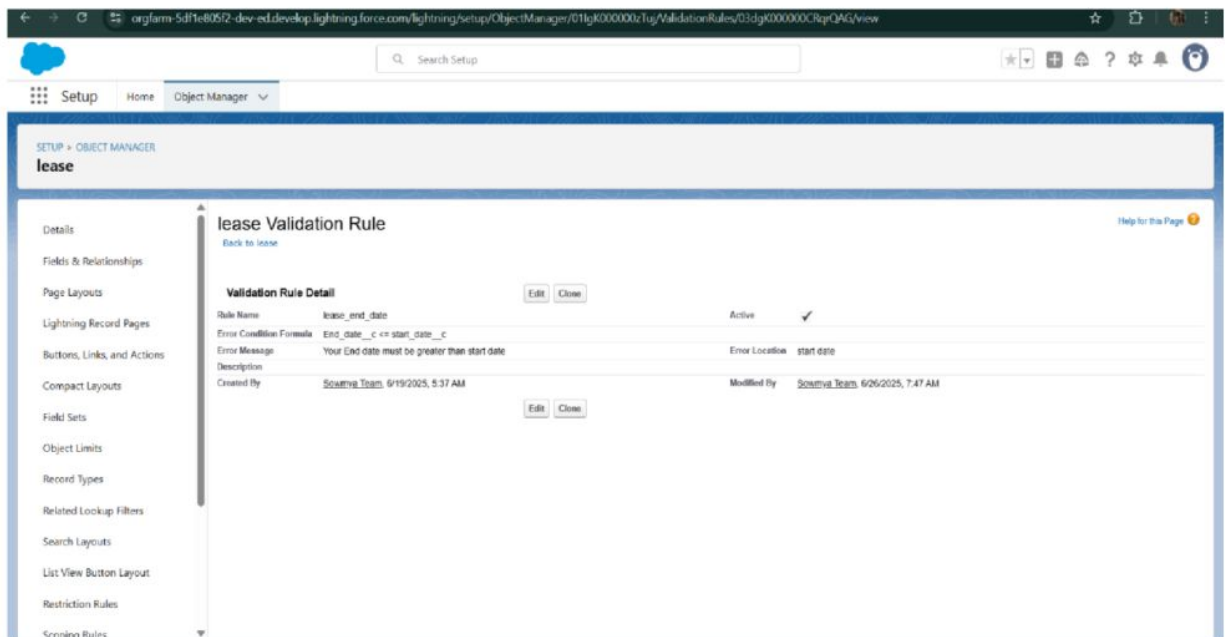
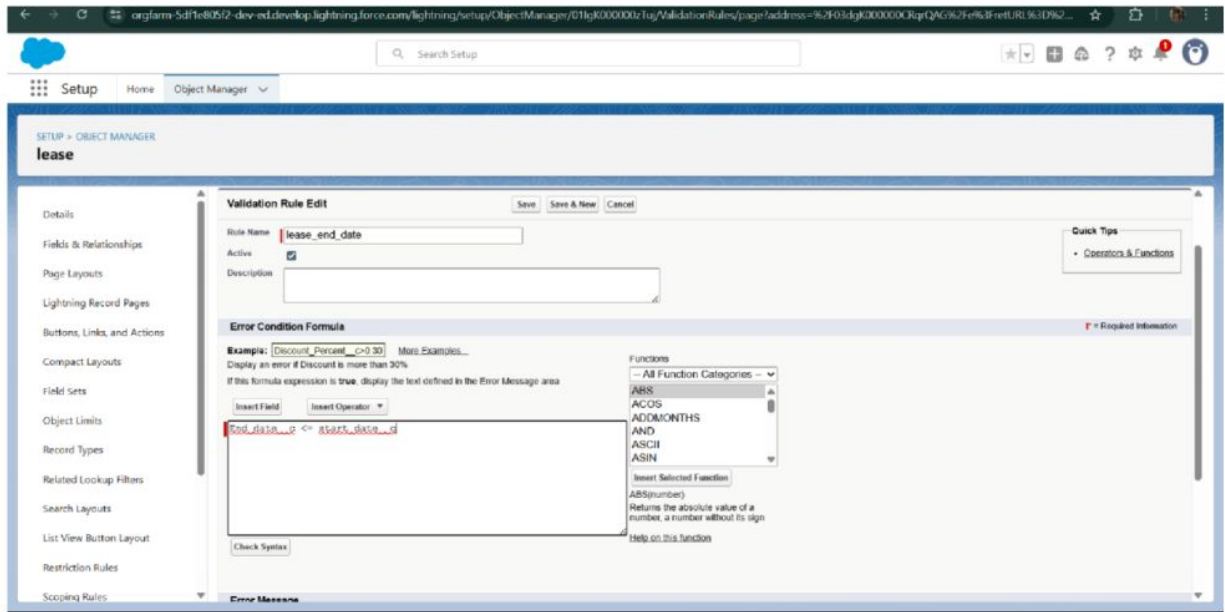




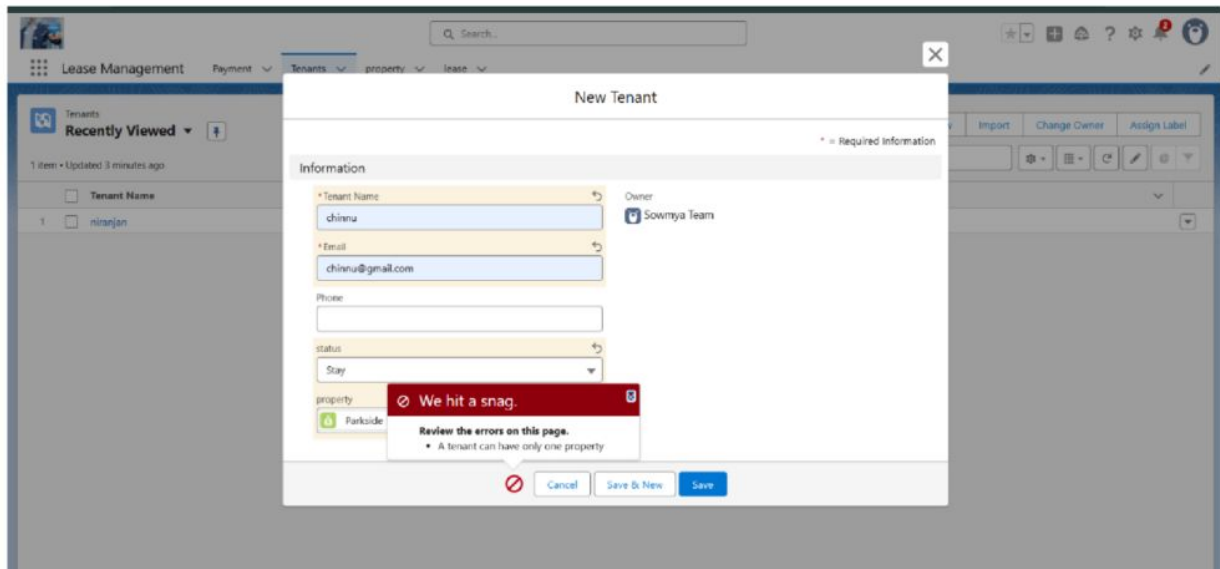
- Implemented Flows for monthly rent and payment success



- To create a validation rule to a Lease Object



- Added Apex trigger to restrict multiple tenants per property



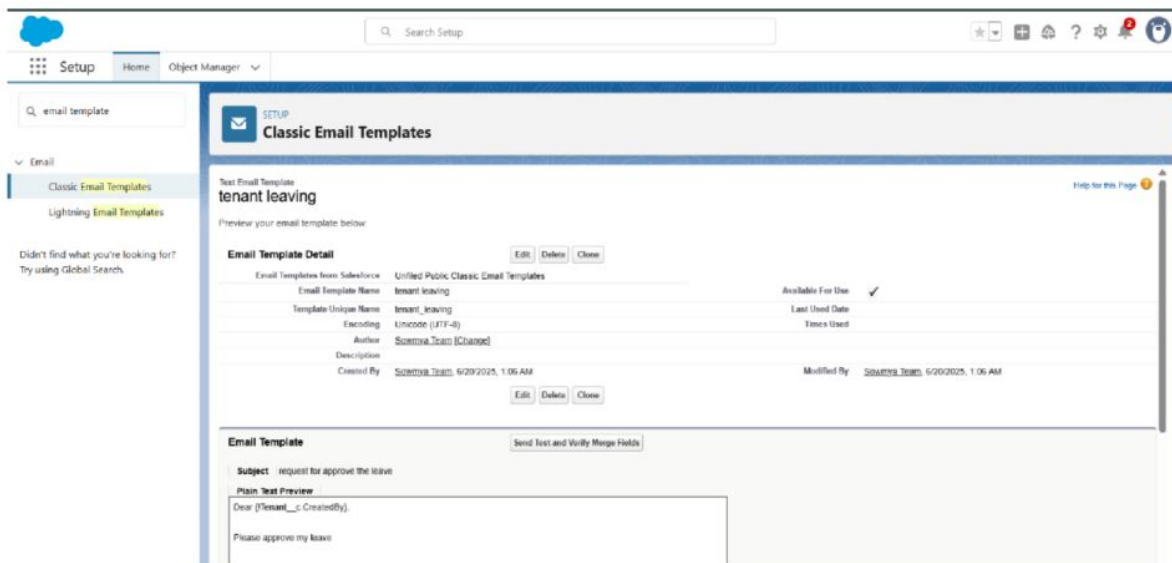
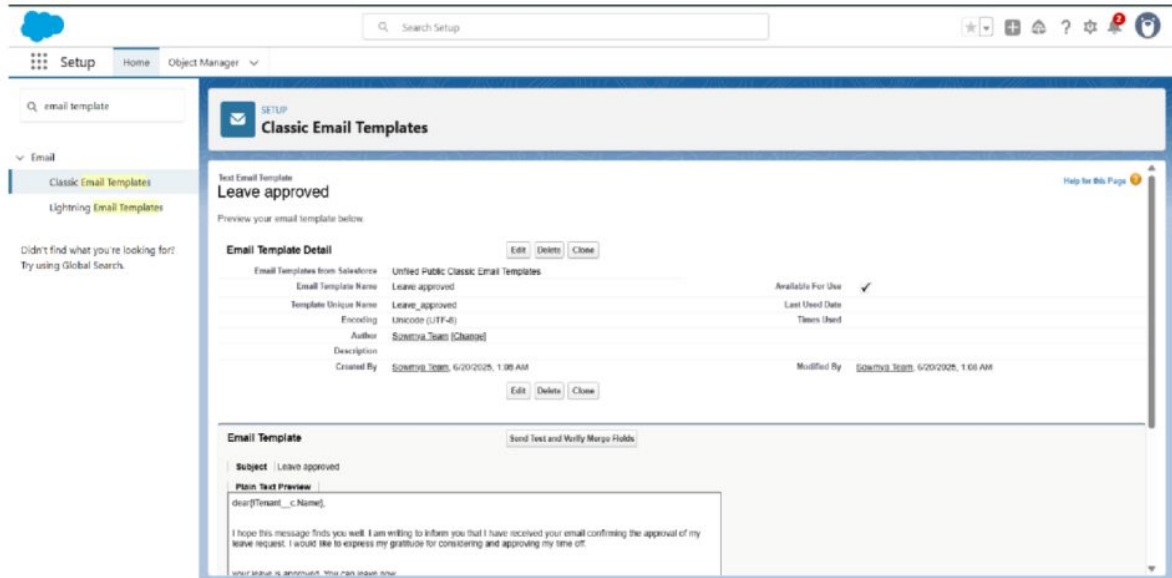
- Scheduled monthly reminder emails using Apex class


```

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 o
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





Search Setup

Setup Home Object Manager

Q email template

Email

- Classic Email Templates
- Lightning Email Templates

Didn't find what you're looking for?  
Try using Global Search.

SETUP

Classic Email Templates

Text Email Template

Leave rejected

Preview your email template below.

Email Template Detail

Email Templates from Salesforce

Email Template Name

Template Unique Name

Encoding

Author

Description

Created By

Unified Public Classic Email Templates

Leave\_rejected

Unicode (UTF-8)

ScoutML Team (Change)

6/20/2025, 1:11 AM

Available For Use

Last Used Date

Times Used

Modified By

ScoutML Team, 6/20/2025, 1:11 AM

Edit

Delete

Clone

Email Template

Send Test and Verify Merge Fields


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



Search Setup

Setup Home Object Manager

Q email template

Email

- Classic Email Templates
- Lightning Email Templates

Didn't find what you're looking for?  
Try using Global Search.

SETUP

Classic Email Templates

Text Email Template

Tenant Email

Preview your email template below.

Email Template Detail

Email Templates from Salesforce

Email Template Name

Template Unique Name

Encoding

Author

Description

Created By

Unified Public Classic Email Templates

Tenant\_Email

Unicode (UTF-8)

ScoutML Team (Change)

6/20/2025, 1:12 AM

Available For Use

Last Used Date

Times Used

Modified By

ScoutML Team, 6/20/2025, 1:12 AM

Edit

Delete

Clone

Email Template

Send Test and Verify Merge Fields

Subject

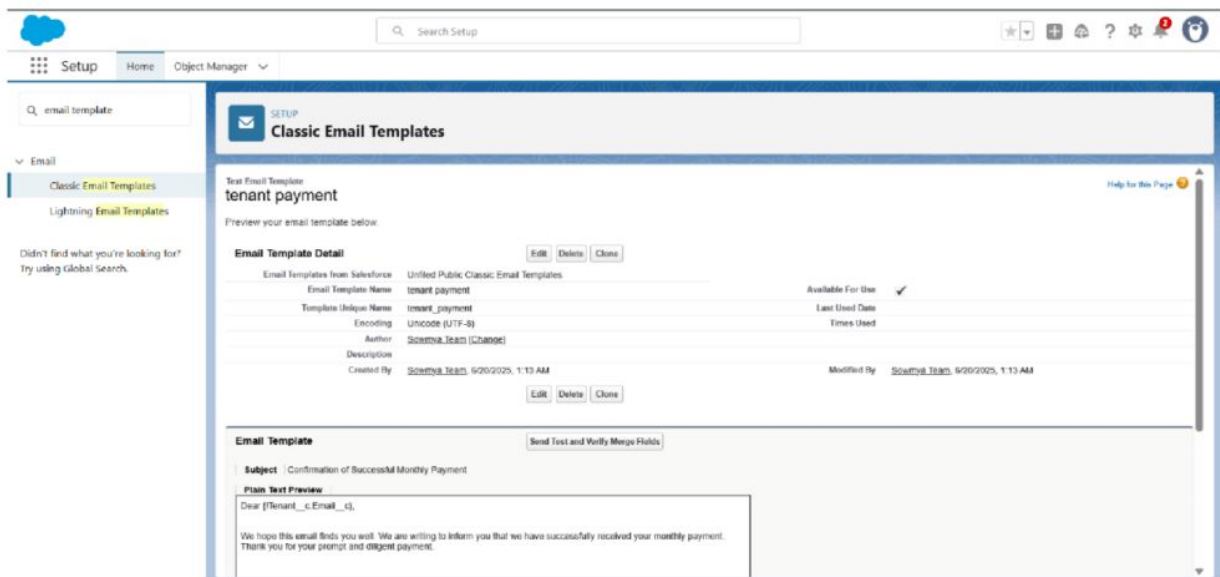
Urgent: Monthly Rent Payment Reminder

Plain Text Preview

Dear {{Tenant\_\_c.Name}},

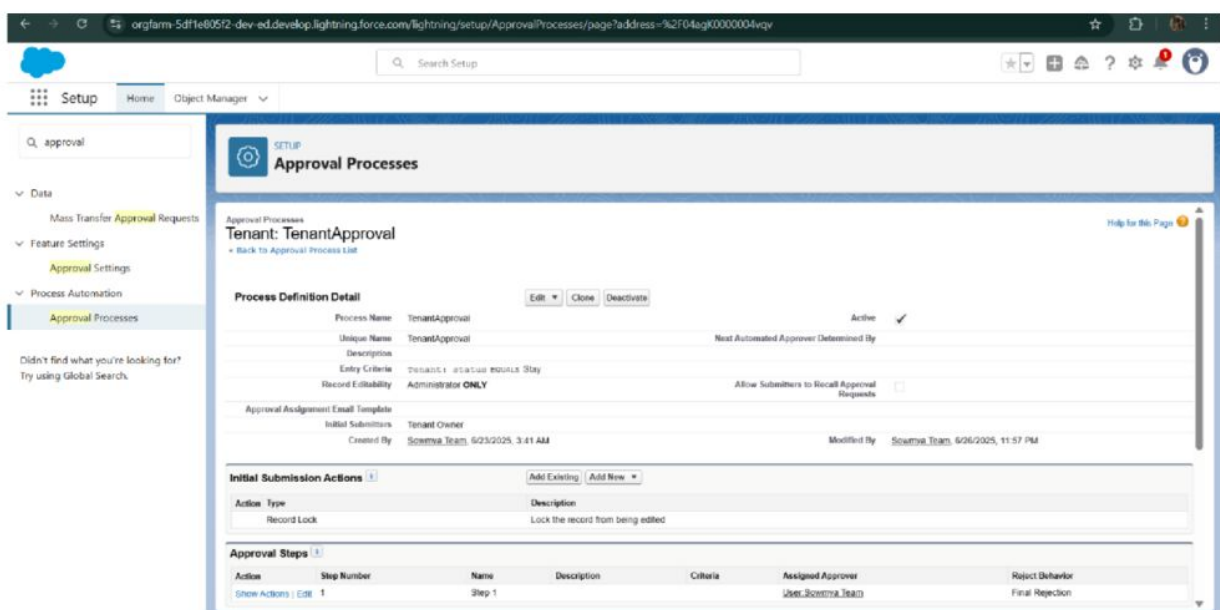
I trust this email finds you well. We appreciate your continued tenancy at our property and I hope you have been comfortable in your residence.

This memorandum is a legally computer generated email generated using equipment, systems, or processes not reviewed, set, reviewed or

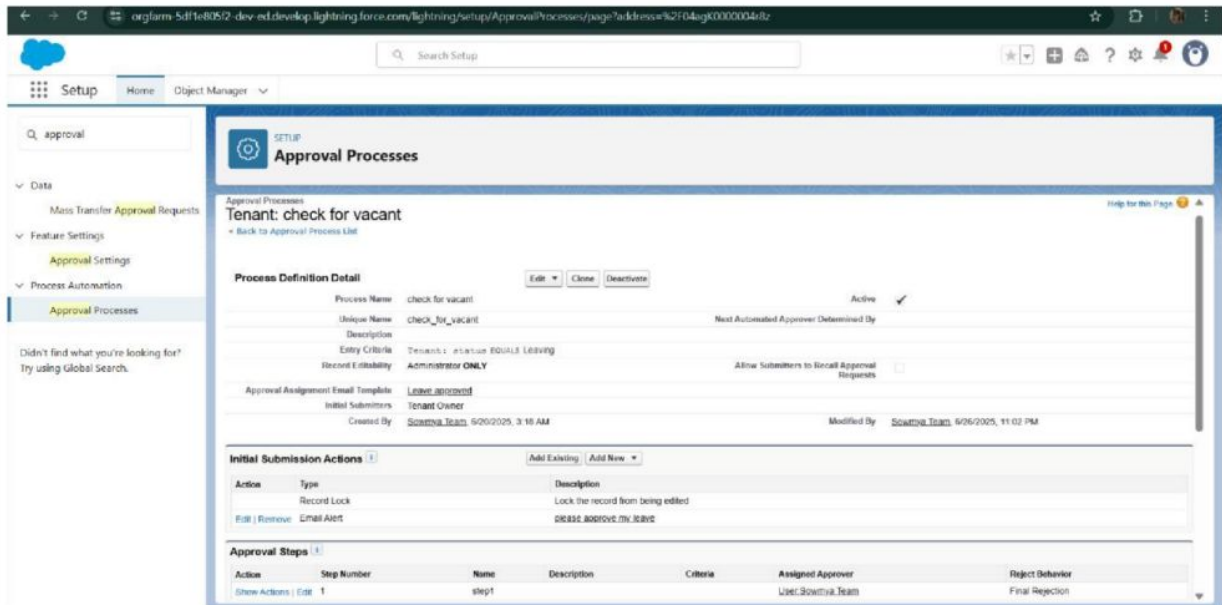


- Approval Process creation

For Tenant Leaving:

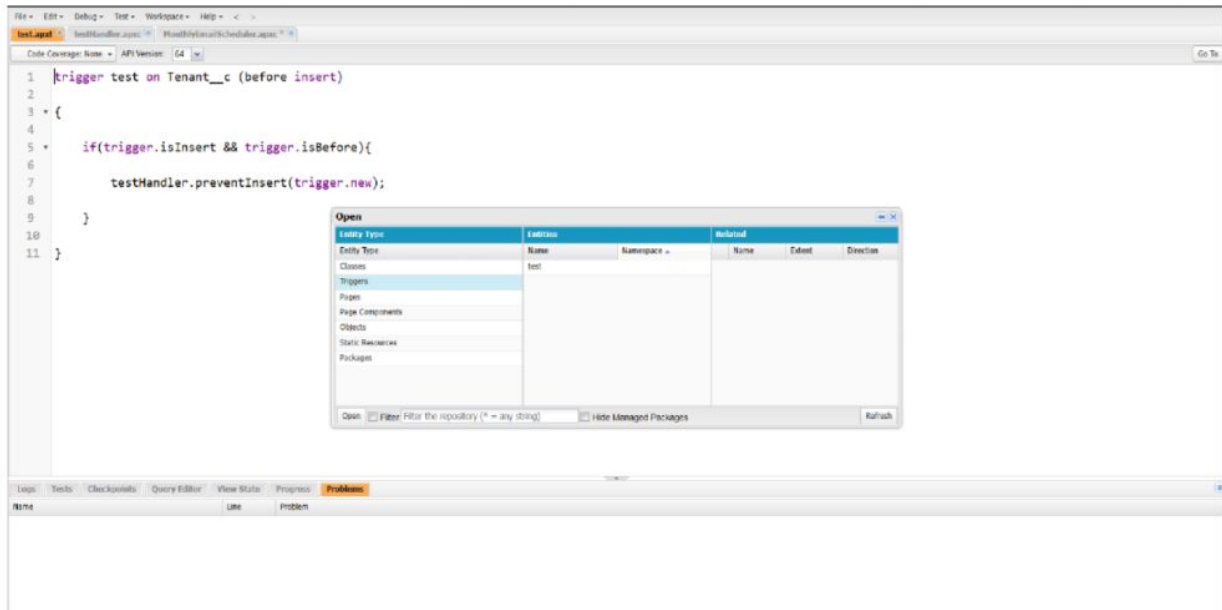


For Check for Vacant:

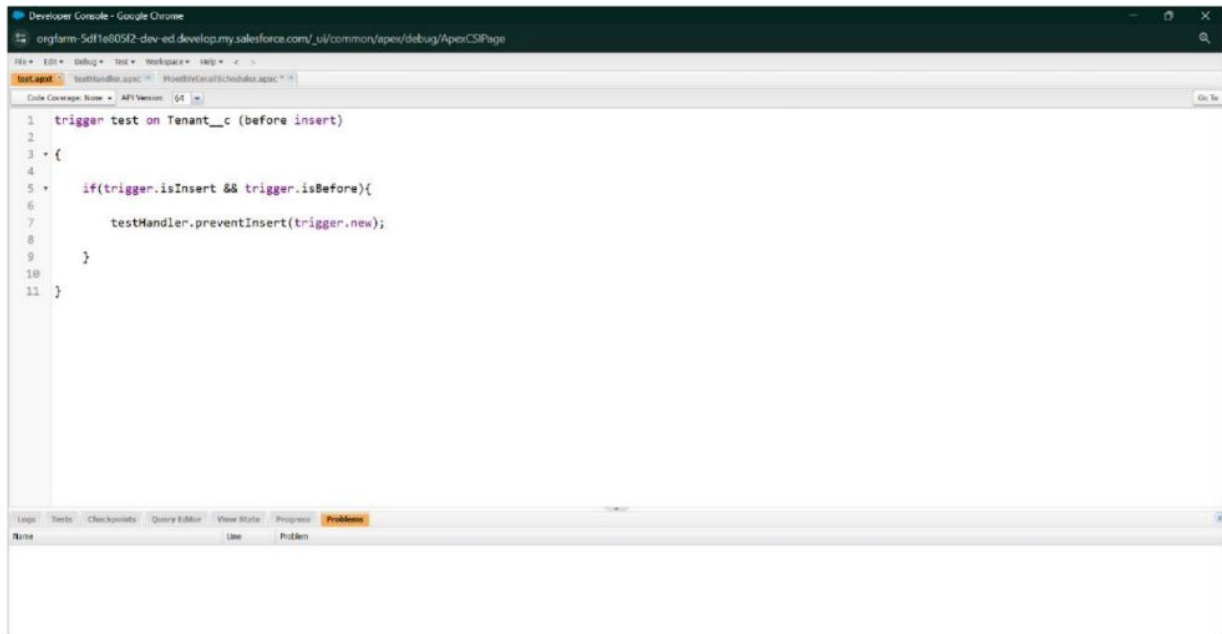


- Apex Trigger

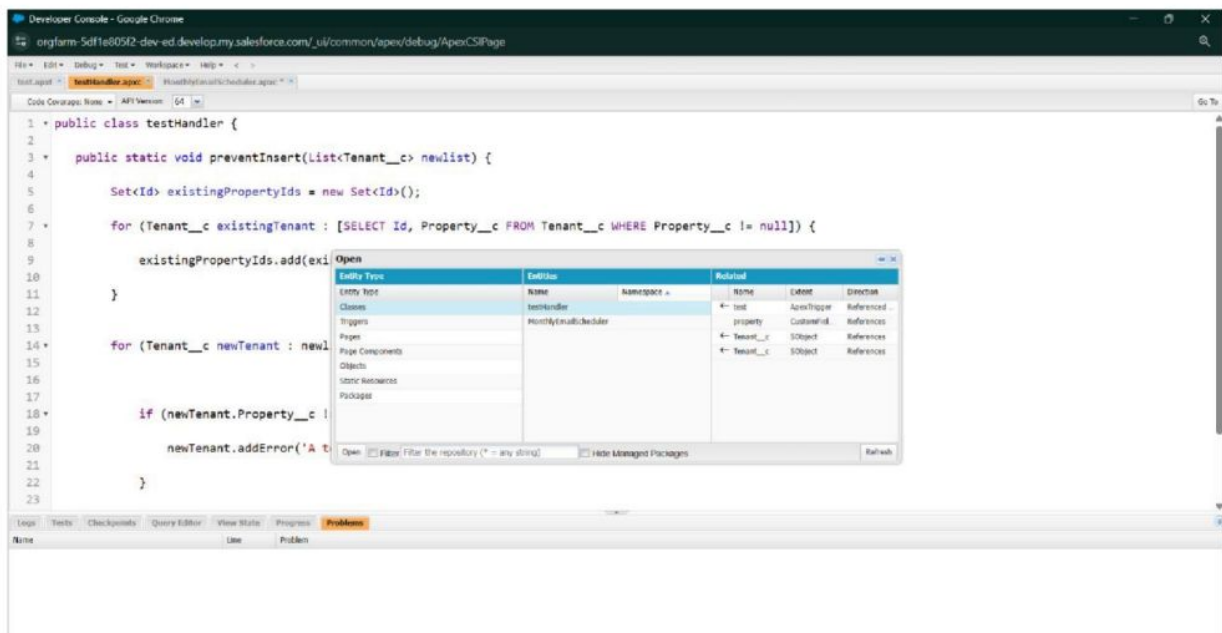
## Create an Apex Trigger







Create an Apex Handler class

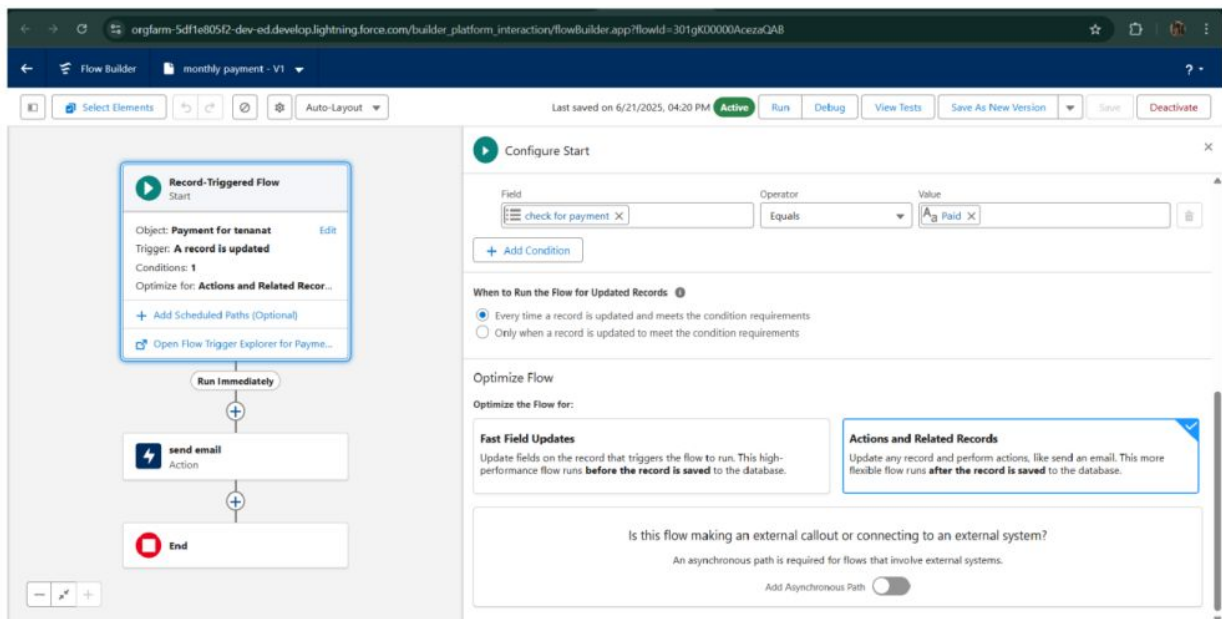


```
Developer Console - Google Chrome
orgfarm-5df1e805f2-dev-ed.develop.my.salesforce.com/ ui/common/apex/debug/ApexCSPage

test.apex? : testHandler.apex : monthlytenantScheduler.apex
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         for (Tenant__c newTenant : newList) {
14
15             if (newTenant.Property__c != null && existingPropertyIds.contains(newTenant.Property__c)) {
16
17                 newTenant.addError('A tenant can have only one property');
18
19             }
20
21         }
22
23     }
24 }
```

## • FLOWS



The screenshot shows the Salesforce Flow Builder interface. On the left, a canvas displays a flow diagram: a 'Record-Triggered Flow Start' block (Object: Payment for tenant, Trigger: A record is updated, Conditions: 1) leads to a 'Run Immediately' block, which then leads to a 'send email' action block, and finally to an 'End' block. On the right, the 'Configure Start' panel is open. It shows 'Select Object' as 'Payment for tenant' and 'Configure Trigger' with 'Trigger the Flow When:' set to 'A record is updated'. Below this, the 'Set Entry Conditions' section shows 'Condition Requirements' set to 'All Conditions Are Met (AND)'. The bottom of the panel has fields for 'Field', 'Operator', and 'Value'.

- Schedule class:  
Create an Apex Class

The screenshot shows the Salesforce Developer Console with an Apex class named 'MonthlyEmailScheduler' implemented. The code is as follows:

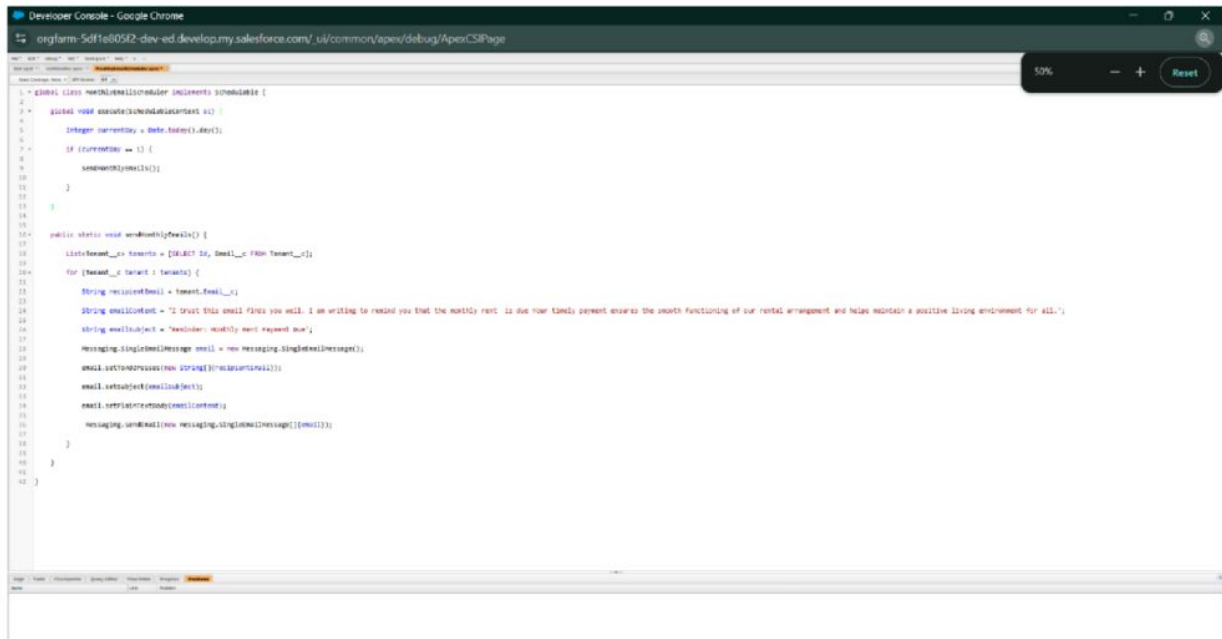
```

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 = [SELE
19
20     for (Tenant__c tenant : tenants
21
22         String recipientEmail = tenant.Email__c;
23

```

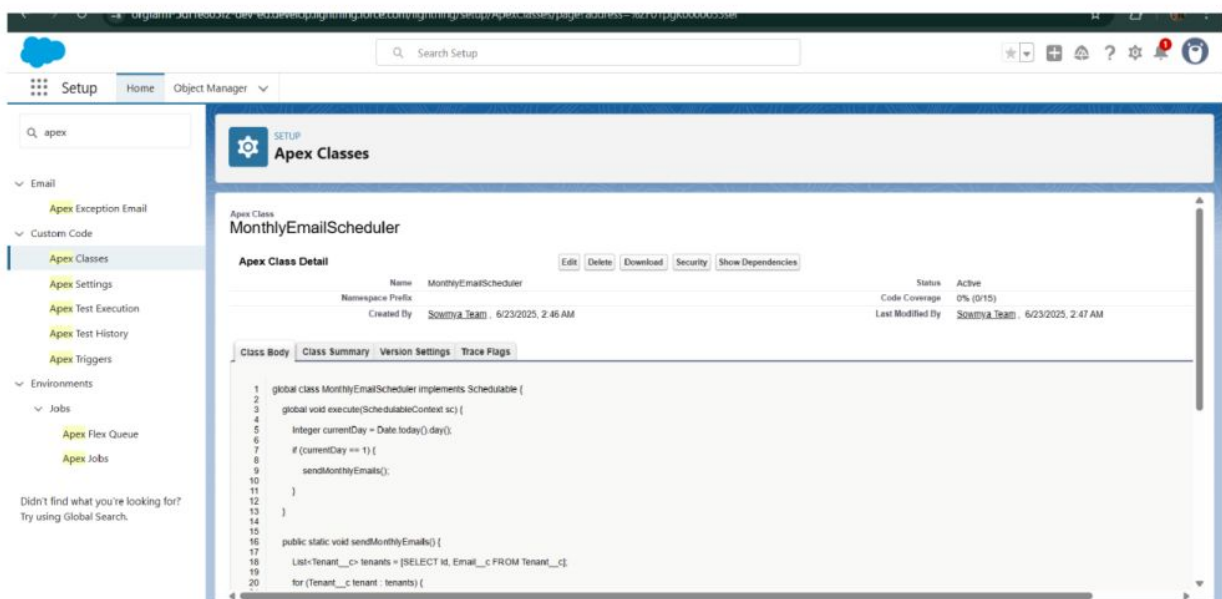
An 'Open' dialog box is visible in the center, showing a table of related items:

Entry Type	Name	Namespace	Related	Field	Details
Class	MonthlyEmailScheduler		CustomTrigger	CustomField	Reference
Trigger	MonthlyEmailScheduler		CustomTrigger	CustomField	Reference
Page	MonthlyEmailScheduler		CustomTrigger	CustomField	Reference
Page Component	MonthlyEmailScheduler		CustomTrigger	CustomField	Reference
Object	MonthlyEmailScheduler		CustomTrigger	CustomField	Reference
Static Resource	MonthlyEmailScheduler		CustomTrigger	CustomField	Reference
Package	MonthlyEmailScheduler		CustomTrigger	CustomField	Reference



```
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     List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
18     for (Tenant__c tenant : tenants) {
19         String recipientEmail = tenant.Email__c;
20         String emailContent = "Hi! I'm writing to remind you that the monthly rent is due now! Please ensure the smooth functioning of our rental arrangement and help maintain a positive living environment for all.";
21         String emailSubject = "Reminder: Monthly Rent Payment Due!";
22         Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
23         email.setToAddresses(new String[] {recipientEmail});
24         email.setSubject(emailSubject);
25         email.setBody(emailContent);
26         Messaging.sendEmail(new Messaging.SingleEmailMessage[] {email});
27     }
28 }
29
30 }
```

## Schedule Apex class



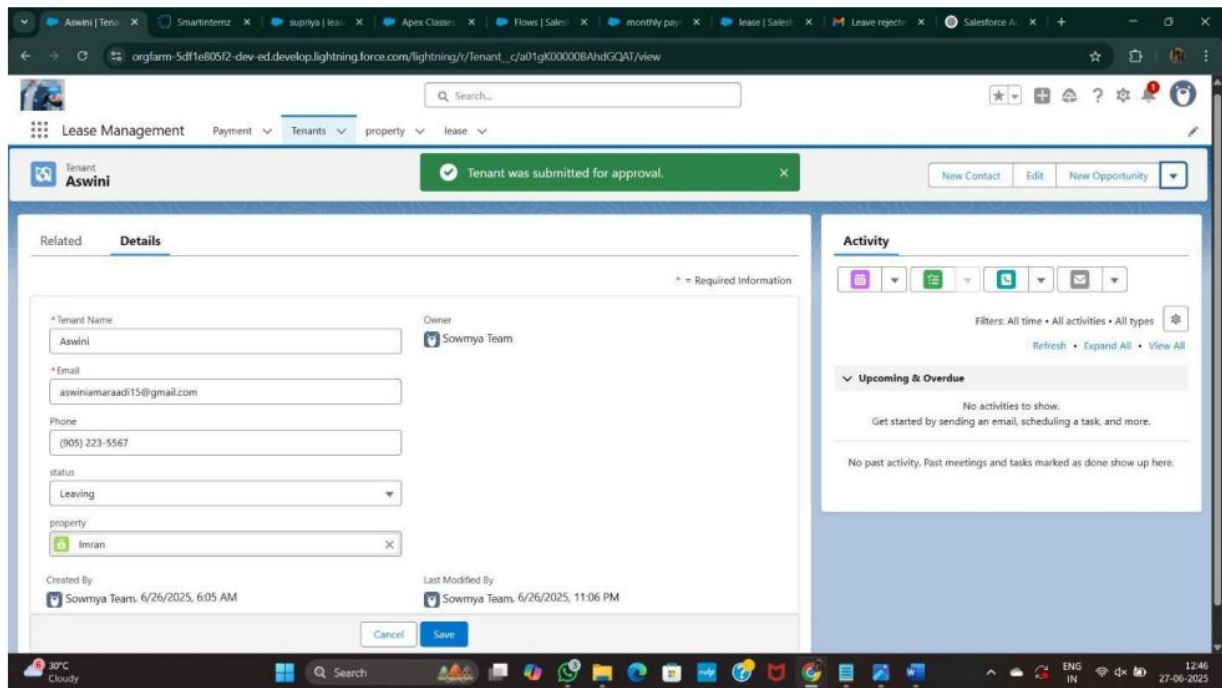
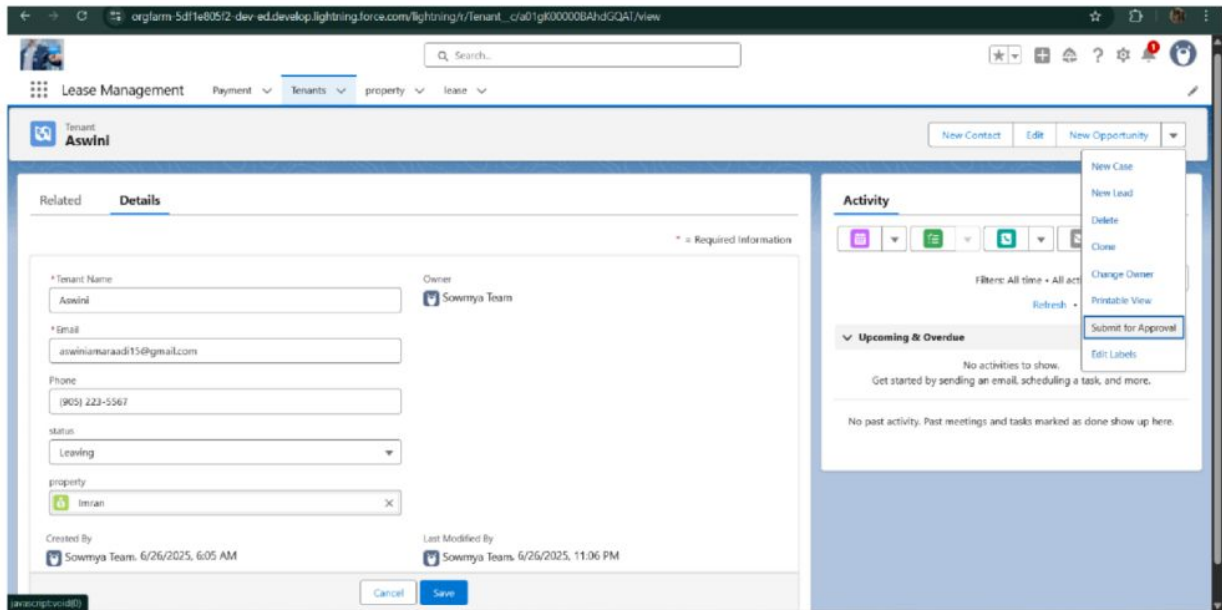
The screenshot shows the Salesforce Setup interface. On the left, the 'Setup' menu is open, and 'Apex Classes' is selected under 'Custom Code'. The main area displays the details for the 'MonthlyEmailScheduler' Apex Class. The class is active, created by 'Somniva Team' on 6/23/2025 at 2:46 AM, and has 0% code coverage. The class body is visible, showing the implementation of the 'execute' method.

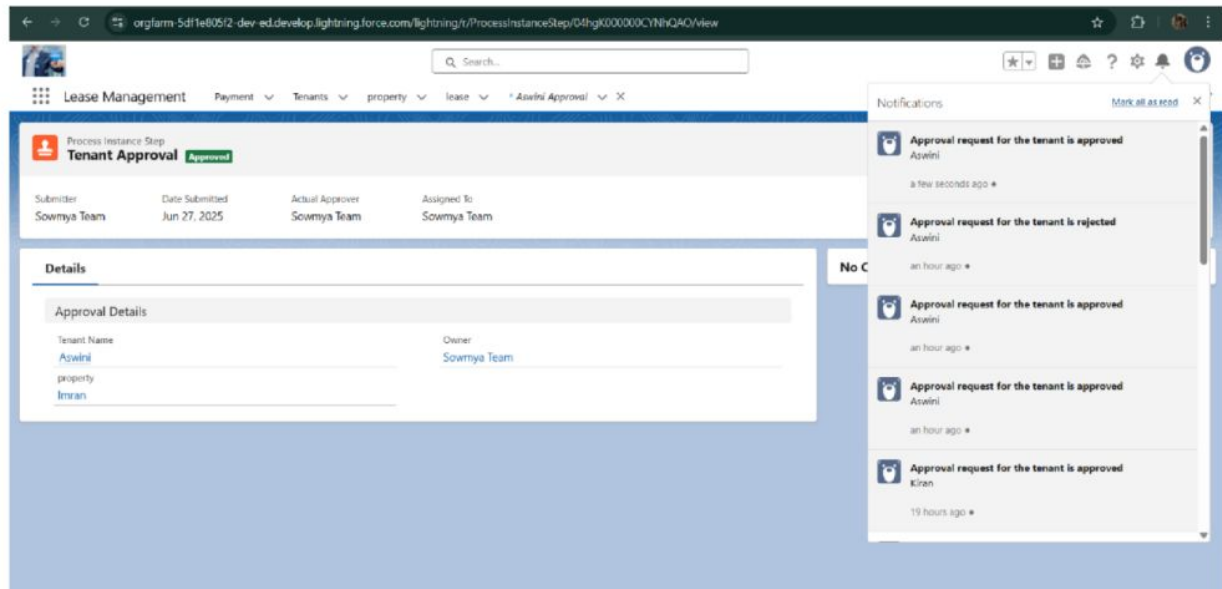
**Apex Class Detail**

Name	MonthlyEmailScheduler	Status	Active
Namespace Prefix		Code Coverage	0% (0/15)
Created By	Somniva Team	Last Modified By	Somniva Team
	6/23/2025, 2:46 AM		6/23/2025, 2:47 AM

**Class Body**

```
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     List<Tenant__c> tenants = [SELECT Id, Email__c FROM Tenant__c];
18     for (Tenant__c tenant : tenants) {
```





# FUNCTIONAL AND PERFORMANCE TESTING

## Performance Testing

- Trigger validation by entering duplicate tenant-property records

Lease Management

Search...

Tenants

New Tenant

\* = Required Information

Information

\*Tenant Name chinu

\*Email chinu@gmail.com

Phone

status Stay

property Parkside

Owner Sowmya Team

We hit a snag.

Review the errors on this page.

- A tenant can have only one property

Cancel Save & New Save

- Validation Rule checking

Lease Management

Search...

lease supriya

New Contact Edit New Opportunity

Related Details

\* = Required Information

\*lease Name supriya

Owner Sowmya Team

start date 6/28/2025

Your End date must be greater than start date

End date 6/19/2025

property Search property...

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

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

We hit a snag.

Review the following fields

- start\_date

Cancel Save

Activity

Filters: All time • All activities • All types

Refresh Expand All View All

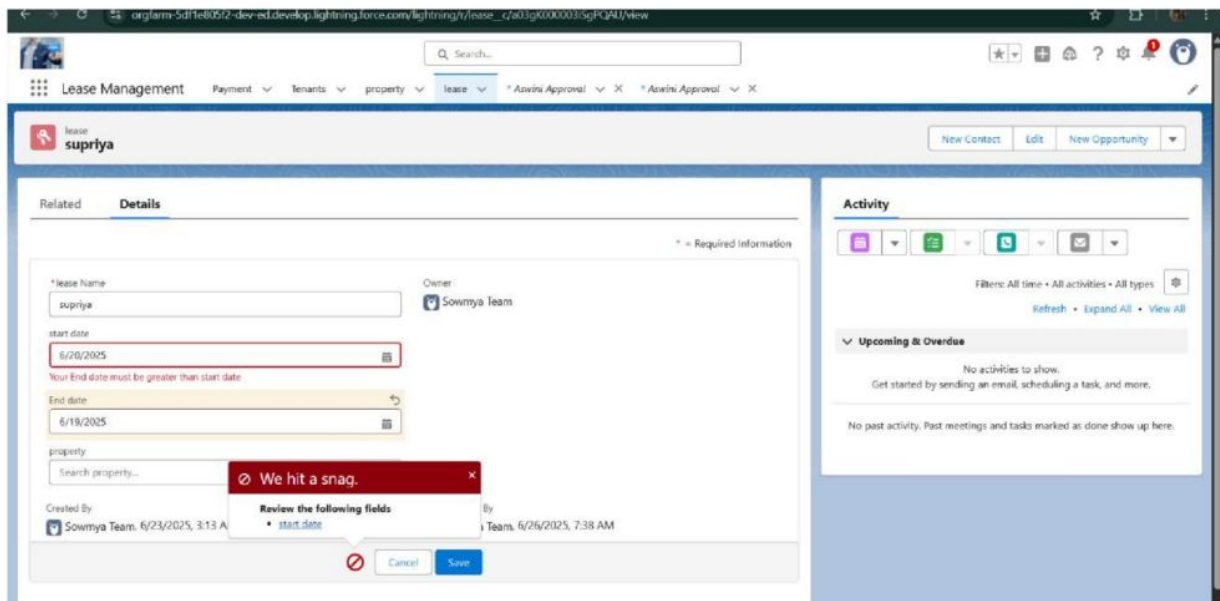
Upcoming & Overdue

No activities to show.

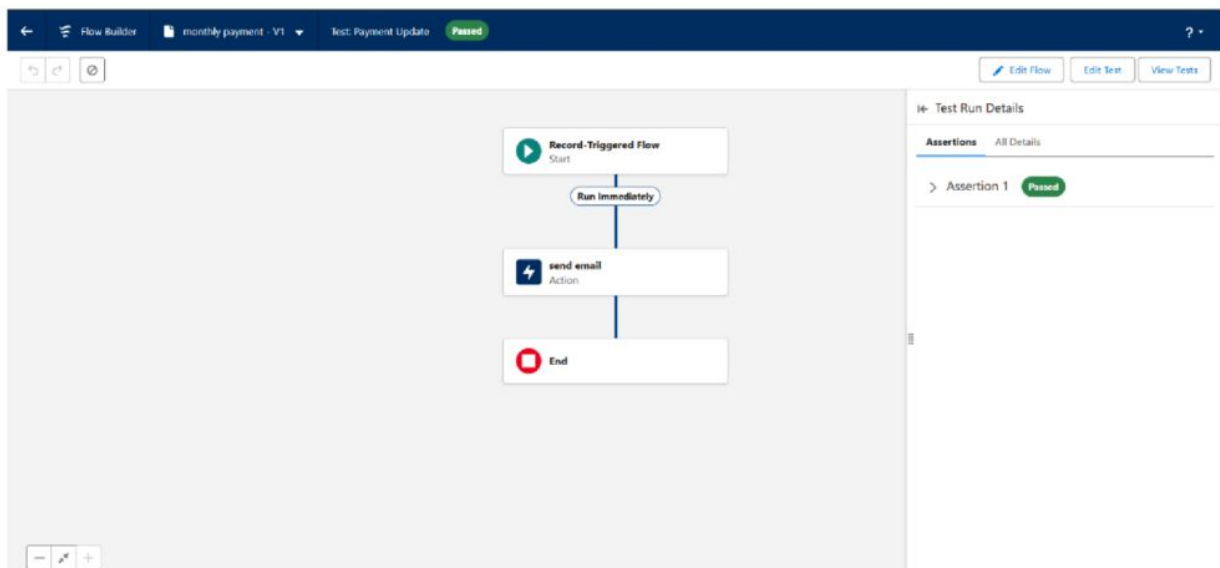
Get started by sending an email, scheduling a task, and more.

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





- Test flows on payment update



- Approval process validated through email alerts and status updates

Lease Management

Payment

Tenants

property

lease

niranjan Approval

Search...

Tenant: niranjan

Related

Details

\* Required Information

Tenant Name

niranjan

Owner

Sowmya Team

Email

niranjan1506@gmail.com

Phone

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

Cancel

Save

Notifications

Mark all as read

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, 1:08 PM

Lease Management

Payment

Tenants

property

lease

Search...

Tenant: niranjan

New Contact

Edit

New Opportunity

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

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

Approval History (6+)

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

View All

Payment (2)

New

Payment Name

Jack

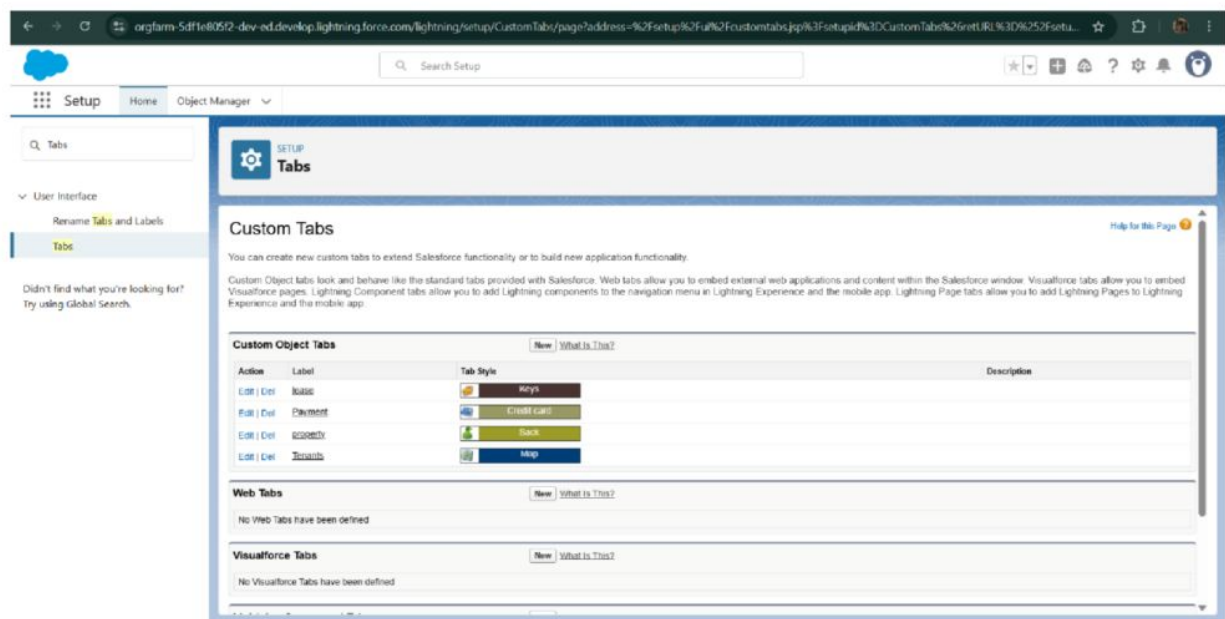
Rahul

View All

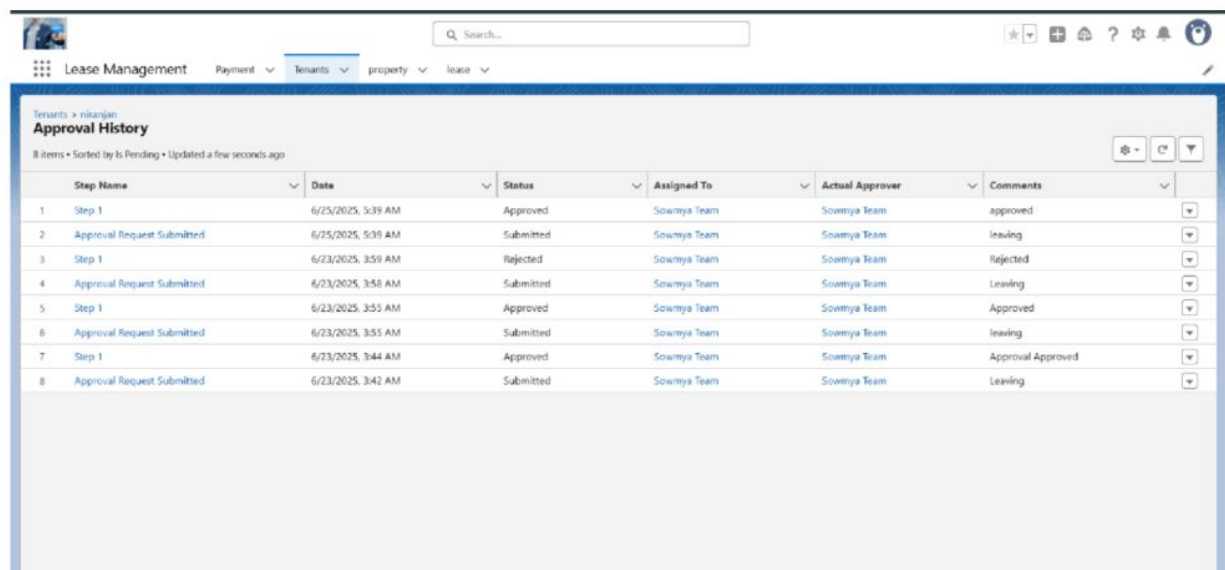
# RESULTS

## Output Screenshots

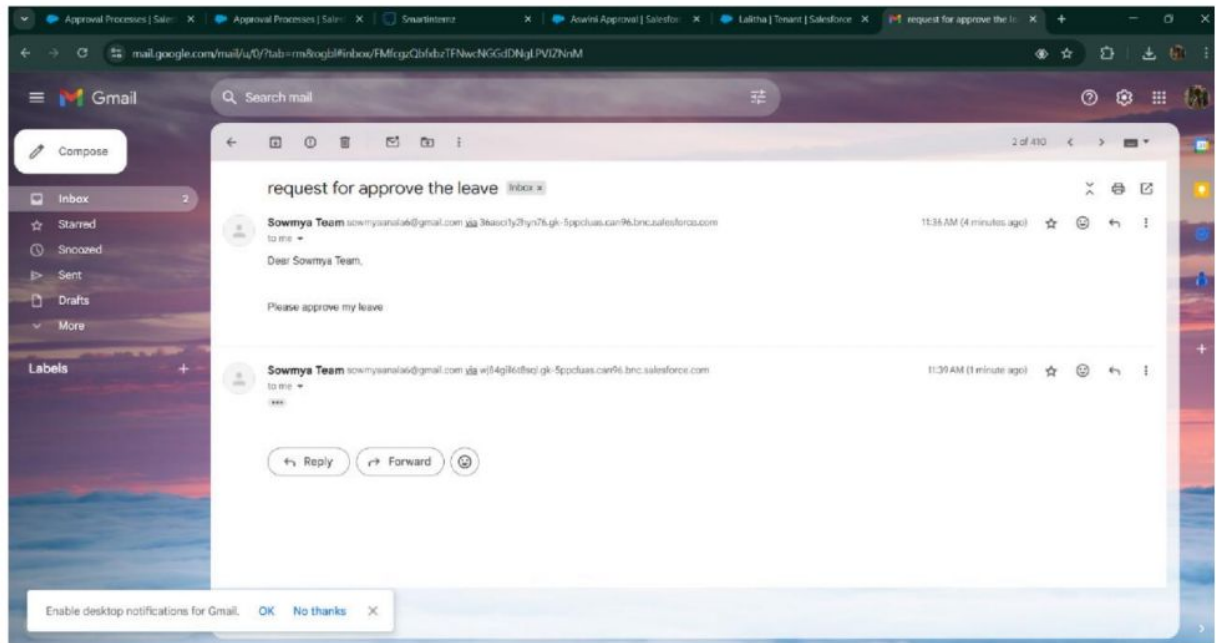
- Tabs for Property, Tenant, Lease, Payment



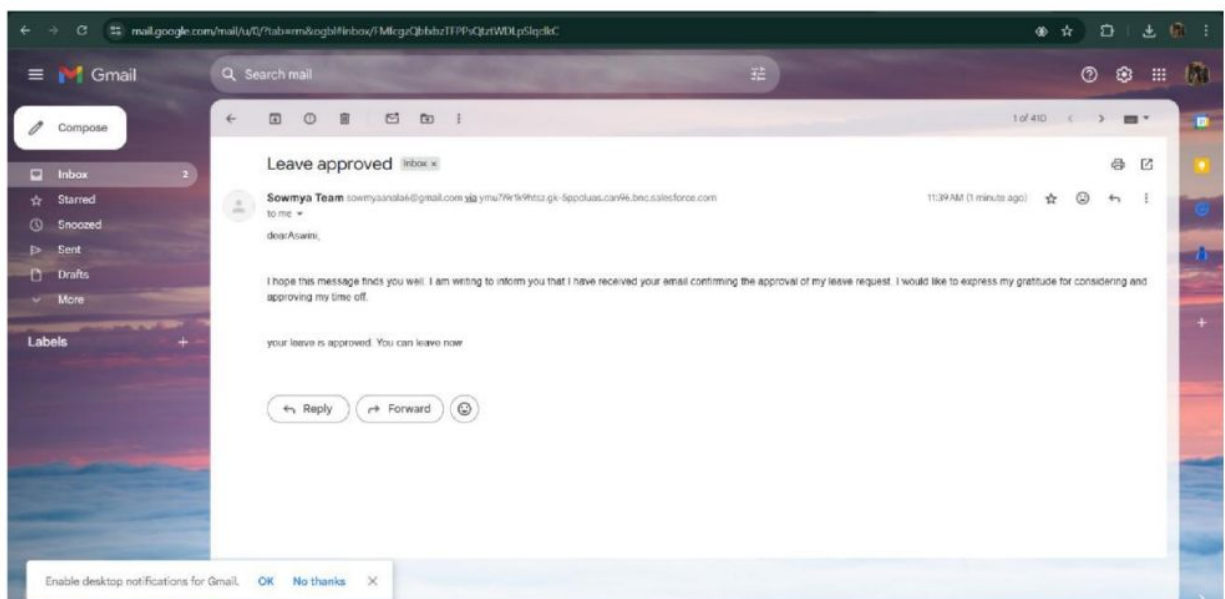
- Email alerts



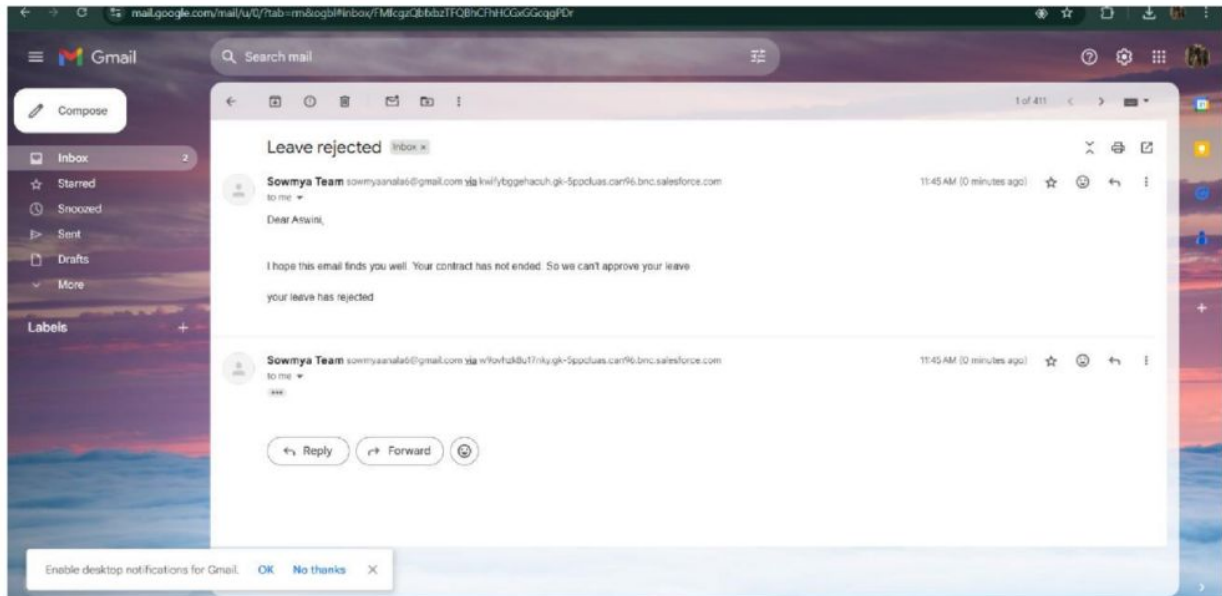
- Request for approve the leave



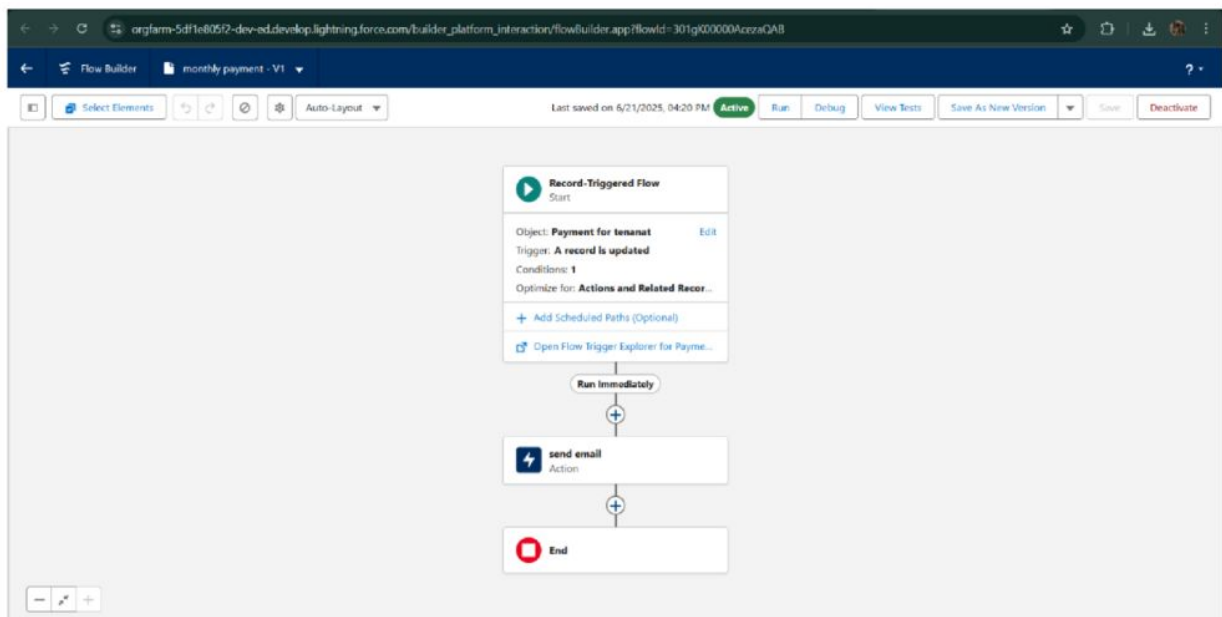
- Leave approved



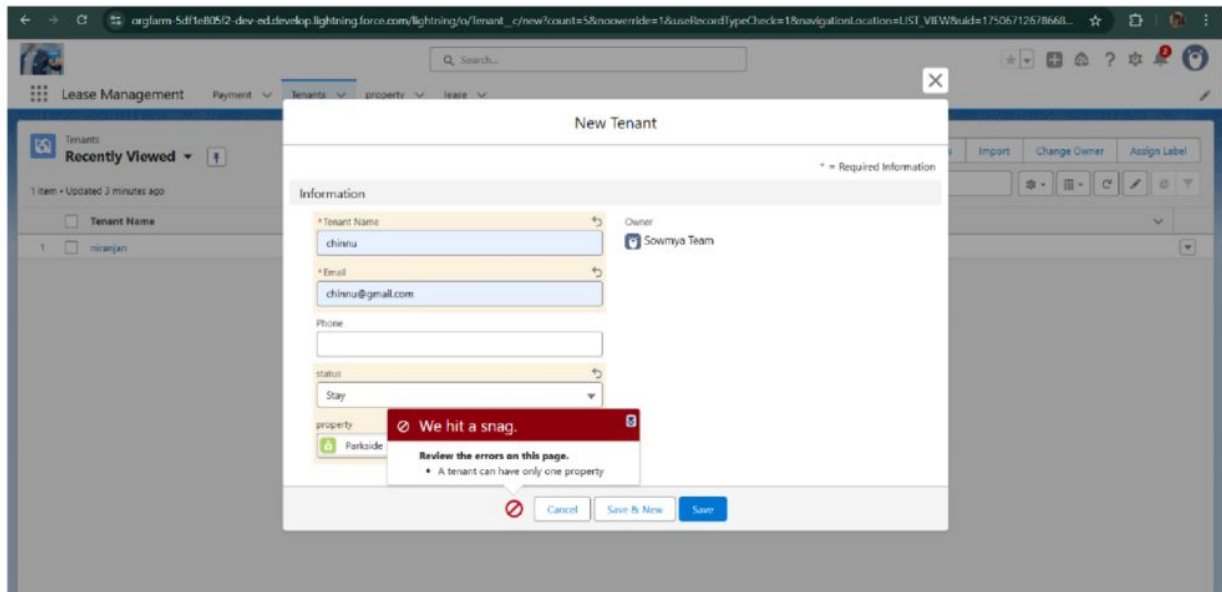
- Leave rejected



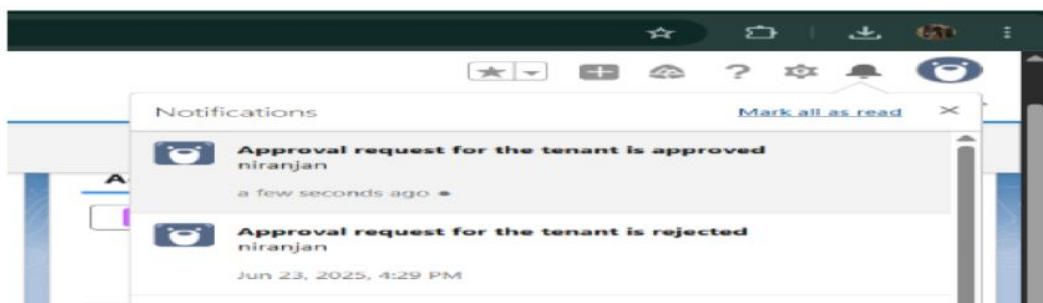
- Flow runs



- 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');

    }

}

}
}

```

### **MonthlyEmailScheduler.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});
}
}
}
```