

# LEASE MANAGEMENT

**College Name: KG college of arts and science**

**TEAM ID: NM2025TMID23722**

**TEAM MEMBERS:**

**Team Leader Name:** DEEPAKKUMAR R

**Email:** 2326ka09@kgcas.com

**Team Member1:** DEIVEEK KRISHNAN N

**Email:** 2326ka10@kgcas.com

**Team Member:** DEVIPRIYA R

**Email:** 2326ka11@kgcas.com

**Team Member:** DIVYASREE M

**Email:** 2326ka12@kgcas.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.



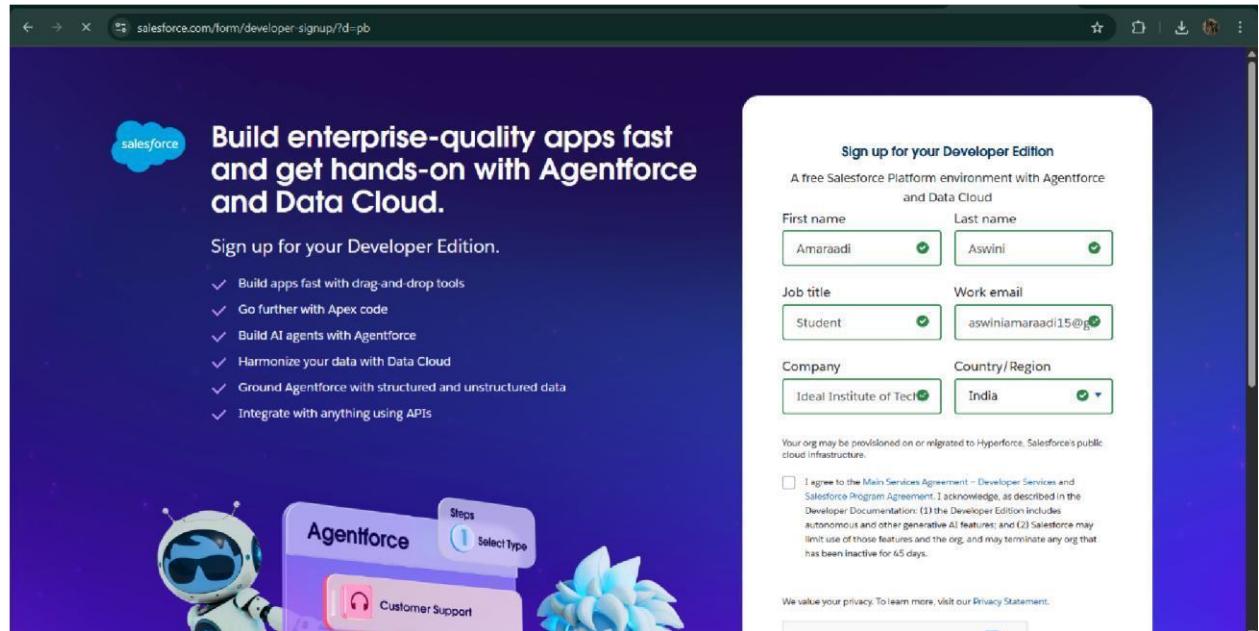
## 1.2 Purpose

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

orgfarm-5dfe805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgk00000zTbN/Details/view

The screenshot shows the Salesforce Object Manager interface. The left sidebar lists various configuration options: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Restriction Rules, and Scoping Rules. The main panel displays the 'Tenant' object's details. The 'Details' section includes fields for Description, API Name (set to 'Tenant\_\_c'), Custom status (checked), Singular Label ('Tenant'), and Plural Label ('Tenants'). On the right, there are sections for Enable Reports (checked), Track Activities (checked), Track Field History (checked), Deployment Status (set to 'Deployed'), and Help Settings (link to 'Standard salesforce.com Help Window'). At the top right are 'Edit' and 'Delete' buttons.

orgfarm-5dfe805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgk00000zTuJ/Details/view

The screenshot shows the Salesforce Object Manager interface. The left sidebar lists various configuration options: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, Restriction Rules, and Scoping Rules. The main panel displays the 'lease' object's details. The 'Details' section includes fields for Description, API Name (set to 'lease\_\_c'), Custom status (checked), Singular Label ('lease'), and Plural Label ('lease'). On the right, there are sections for Enable Reports (checked), Track Activities (checked), Track Field History (checked), Deployment Status (set to 'Deployed'), and Help Settings (link to 'Standard salesforce.com Help Window'). At the top right are 'Edit' and 'Delete' buttons.

**Object Manager**

**Payment for tenantat**

**Details**

Description

API Name: Payment\_for\_tenantat\_c

Custom

Singular Label: Payment for tenantat

Plural Label: Payment

Enable Reports: ✓

Track Activities: ✓

Track Field History: ✓

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

- Configured fields and relationships

**Fields & Relationships**

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address__c	Long Text Area(32768)		
Created By	CreatedBy	Lookup(User)		
Last Modified By	LastModifiedBy	Lookup(User)		
Name	Name__c	Text(25)		
Owner	OwnerId	Lookup(User,Group)		✓
property	property_c	Lookup(property)		✓
property Name	Name	Text(80)		✓
sfqt	sfqt_c	Text(18)		
Type	Type__c	Picklist		

Setup > Object Manager  
Payment for tenantat

Fields & Relationships		Fields & Relationships																																								
		FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED																																				
<a href="#">Page Layouts</a> <a href="#">Lightning Record Pages</a> <a href="#">Buttons, Links, and Actions</a> <a href="#">Compact Layouts</a> <a href="#">Field Sets</a> <a href="#">Object Limits</a> <a href="#">Record Types</a> <a href="#">Related Lookup Filters</a> <a href="#">Search Layouts</a> <a href="#">List View Button Layout</a> <a href="#">Restriction Rules</a>	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)			
	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		Fields & Relationships																																								
		FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED																																				
<a href="#">Page Layouts</a> <a href="#">Lightning Record Pages</a> <a href="#">Buttons, Links, and Actions</a> <a href="#">Compact Layouts</a> <a href="#">Field Sets</a> <a href="#">Object Limits</a> <a href="#">Record Types</a> <a href="#">Related Lookup Filters</a> <a href="#">Search Layouts</a> <a href="#">List View Button Layout</a> <a href="#">Restriction Rules</a> <a href="#">Scoping Rules</a>	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			
	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																																								

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 a 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: [Image Preview]

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**

[App Launcher Preview Image]

Lightning App Builder | App Settings | Pages | Lease Management | ? Help

**App Settings**

App Details & Branding  
App Options  
Utility Items (Desktop Only)

**Navigation Items**

User Profiles

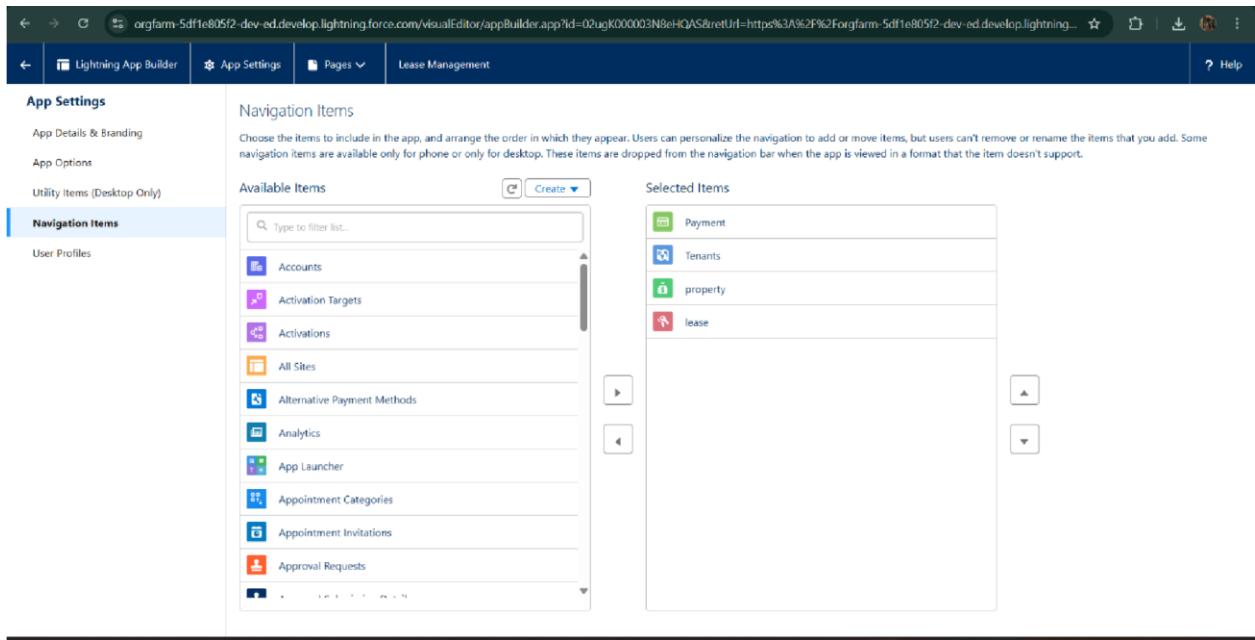
**Available Items**

Type to filter list...

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

**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)

**User Profiles**

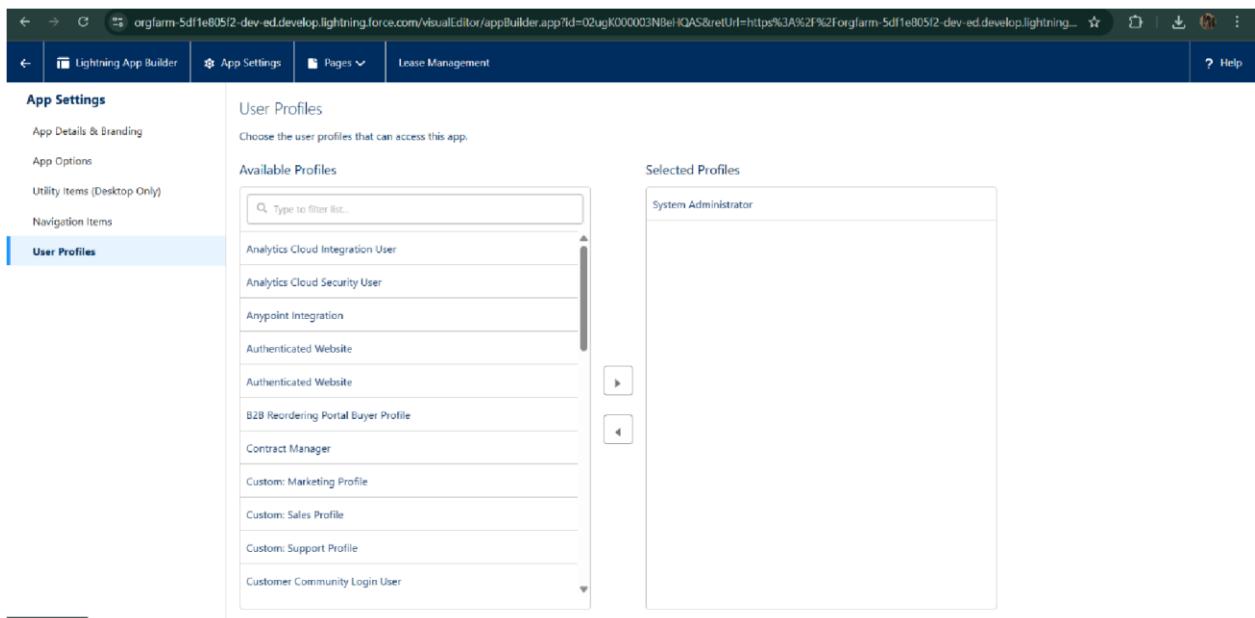
**Available Profiles**

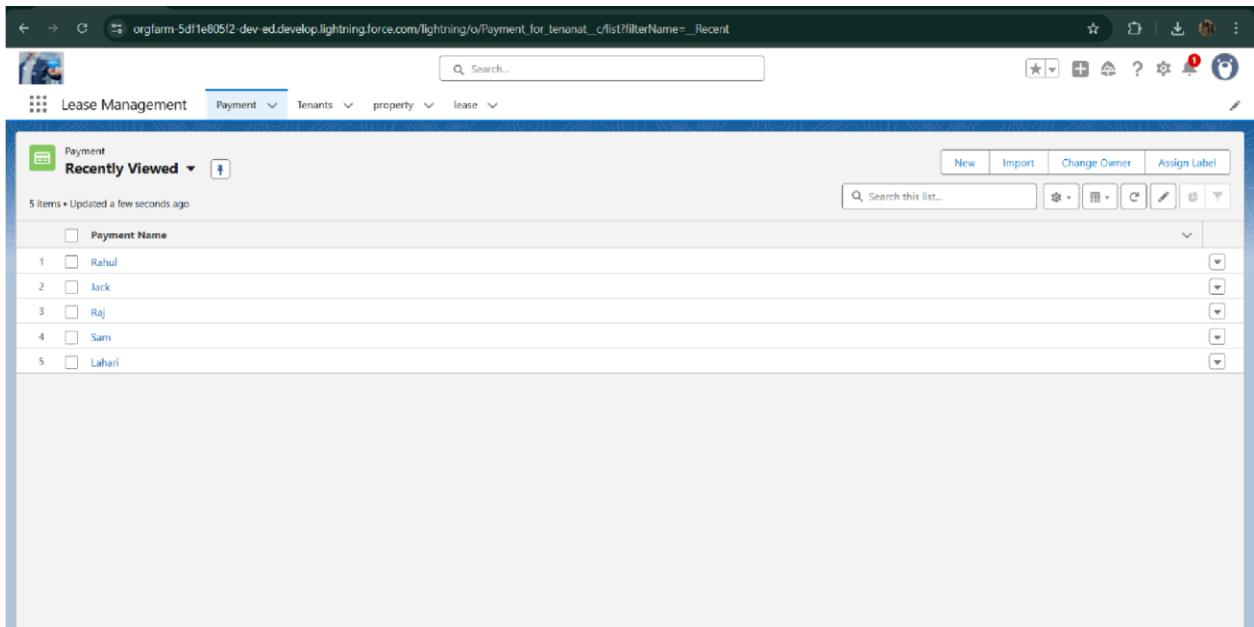
Type to filter list...

- 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

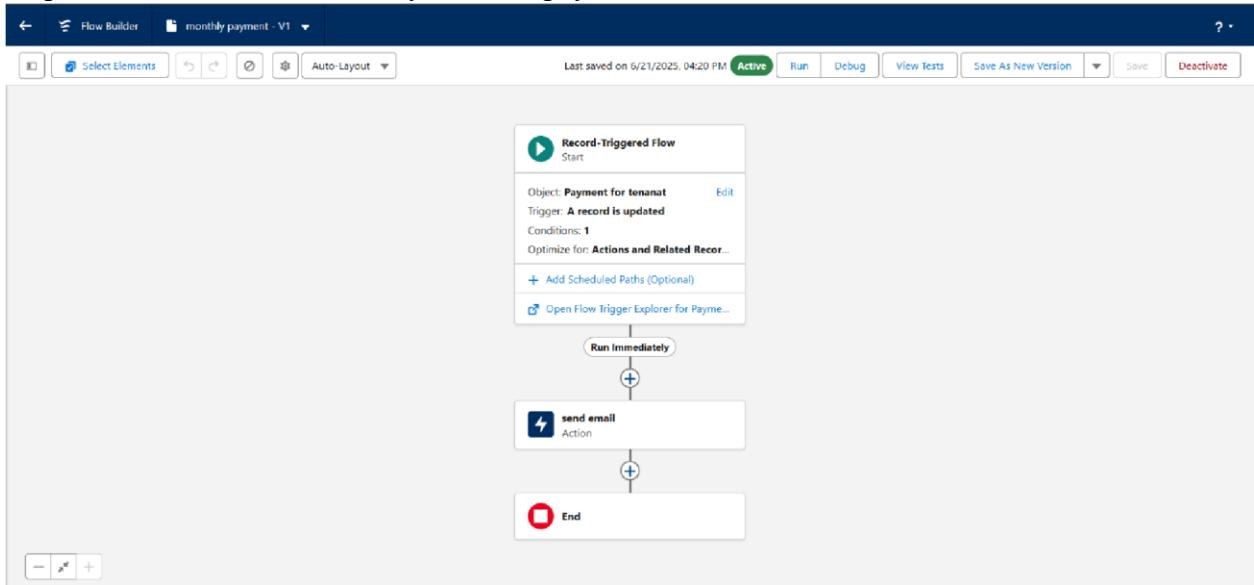
**Selected Profiles**

- System Administrator





- Implemented Flows for monthly rent and payment success



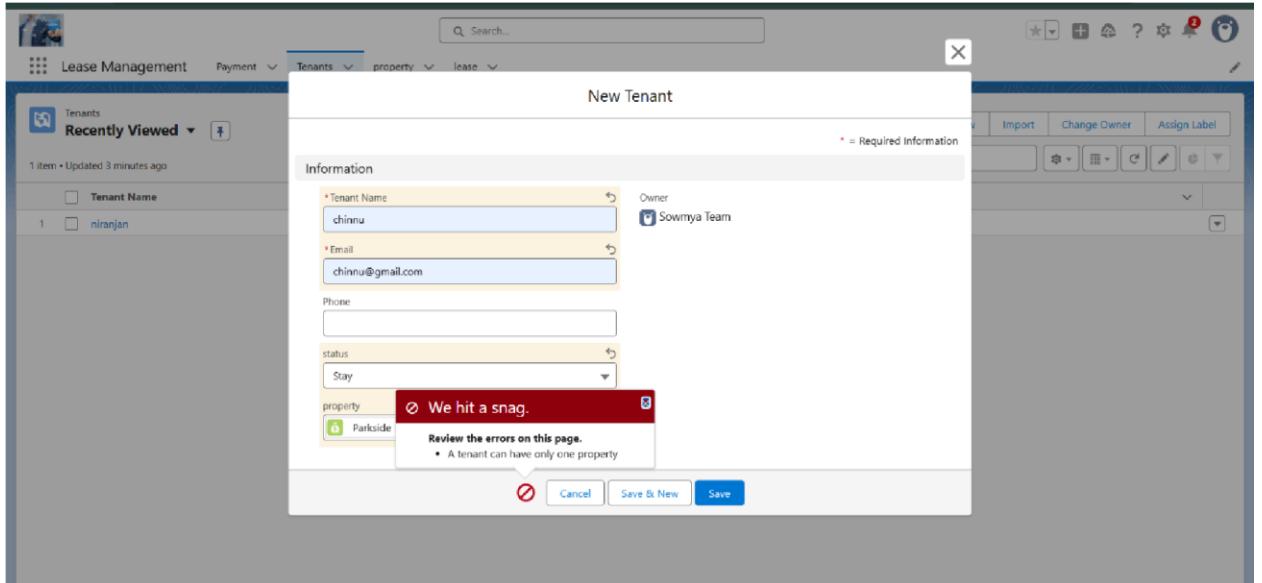
- To create a validation rule to a Lease Object

The screenshot shows the Salesforce Setup interface with the URL <https://orgfarm-5d1fe805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK00000zTuj/ValidationRules/page?address=%2F03dgK00000CRqrQAG%2Fe%3fretURL%3D%2...>. The page title is "Validation Rule Edit" for the "lease" object. The "Rule Name" field contains "lease\_end\_date". The "Active" checkbox is checked. The "Description" field is empty. The "Error Condition Formula" section 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 "Check Syntax" button is visible at the bottom left of the formula editor.

The screenshot shows the Salesforce Setup interface with the URL <https://orgfarm-5d1fe805f2-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01lgK00000zTuj/ValidationRules/03dgK00000CRqrQAG/view>. The page title is "Validation Rule Detail" for the "lease" object. The "Validation Rule Detail" table shows the following information:

Field	Value
Rule Name	lease_end_date
Error Condition Formula	End_date_c <= start_date_c
Error Message	Your End date must be greater than start date
Description	
Created By	Sowmya Team
Modified By	Sowmya Team
Active	✓
Error Location	start_date

- Added Apex trigger to restrict multiple tenants per property



- Scheduled monthly reminder emails using Apex class



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

- Title Bar:** Shows "File -> Edit -> Run -> Test -> Workspaces -> Help -> <".
- Toolbar:** Includes icons for New, Open, Save, Cut, Copy, Paste, Find, Replace, Undo, Redo, and others.
- Code Editor:** Displays Java code for a "MonthlyEmailScheduler" class. The code implements the "Schedulable" interface and performs a scheduled task to send monthly emails to tenants.
- Status Bar:** Shows "Log | Tools | Checkpoints | Query Editor | View Status | Progress | Problems".
- Bottom Bar:** Shows "New | Use | Publish".

```
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 }
```

- 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'. The 'Classic Email Templates' section is selected under the 'Email' category. A specific email template named '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 [Change]
- Description: Created By: Sowmya Team, 6/20/2025, 10:08 AM
- Modified By: Sowmya Team, 6/20/2025, 10:08 AM
- Status: Available For Use (checked)
- Last Used Date: Not specified
- Times Used: Not specified

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

```

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 search bar set to 'email template'. The 'Classic Email Templates' section is selected under the 'Email' category. A specific email template named '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 [Change]
- Description: Created By: Sowmya Team, 6/20/2025, 1:06 AM
- Modified By: Sowmya Team, 6/20/2025, 1:06 AM
- Status: Available For Use (checked)
- Last Used Date: Not specified
- Times Used: Not specified

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

```

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:

- Page Header:** Includes the Salesforce logo, a search bar labeled "Search Setup", and various navigation icons.
- Left Navigation Bar:** Under the "Email" section, "Classic Email Templates" is selected.
- Main Content Area:**
  - Title:** Text Email Template **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	Sowmya_Team [Changed]
Description	
Created By	Sowmya_Team 6/20/2025, 1:11 AM
Modified By	Sowmya_Team 6/20/2025, 1:11 AM
  - Email Template Preview:**
    - 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:

- Page Header:** Includes the Salesforce logo, a search bar labeled "Search Setup", and various navigation icons.
- Left Navigation Bar:** Under the "Email" section, "Classic Email Templates" is selected.
- Main Content Area:**
  - Title:** Text Email Template **Tenant Email**
  - Email Template Detail:**

Email Templates from Salesforce	Unified Public Classic Email Templates
Email Template Name	Tenant Email
Template Unique Name	Tenant_Email
Encoding	Unicode (UTF-8)
Author	Sowmya_Team [Changed]
Description	
Created By	Sowmya_Team 6/20/2025, 1:12 AM
Modified By	Sowmya_Team 6/20/2025, 1:12 AM
  - Email Template Preview:**
    - 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.
```

**Classic Email Templates**

**tenant payment**

**Email Template Detail**

- Email Template Name: tenant payment
- Template Unique Name: tenant\_payment
- Encoding: Unicode (UTF-8)
- Author: Sowmya.Team [Change]
- Created By: Sowmya.Team 6/26/2025, 1:13 AM
- Modified By: Sowmya.Team 6/26/2025, 1:13 AM

**Email Template**

**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:

**Approval Processes**

**Tenant: TenantApproval**

**Process Definition Detail**

- Process Name: TenantApproval
- Unique Name: TenantApproval
- Description:
- Entry Criteria: Tenant\_\_r\_Status EQUALS Stay
- Record Editability: Administrator ONLY
- Active: checked
- Next Automated Approver Determined By:
- Allow Submitters to Recall Approval Requests: unchecked

**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:** Tenant: check for vacant
- Process Definition Detail:**
  - Process Name: check for vacant
  - Unique Name: check\_for\_vacant
  - Description:
  - Entry Criteria: Tenant\_\_status EQUALS Leaving
  - Record Entitlity: Administrator ONLY
  - Next Automated Approver Determined By:
  - Allow Submitters to Recall Approval Requests:
  - Approval Assignment Email Template: Leave approved
  - Initial Submitters: Tenant Owner
  - Created By: Scwmya Team
  - Modified By: Scwmya Team
  - Created Date: 6/20/2025, 3:16 AM
  - Modified Date: 6/26/2025, 11:02 PM
- Initial Submission Actions:**
  - Action: Record Lock
  - Type: Record Lock
  - Description: Lock the record from being edited
  - Email Alert: RELEASE\_APPROVE\_My\_LEAVE
- Approval Steps:**

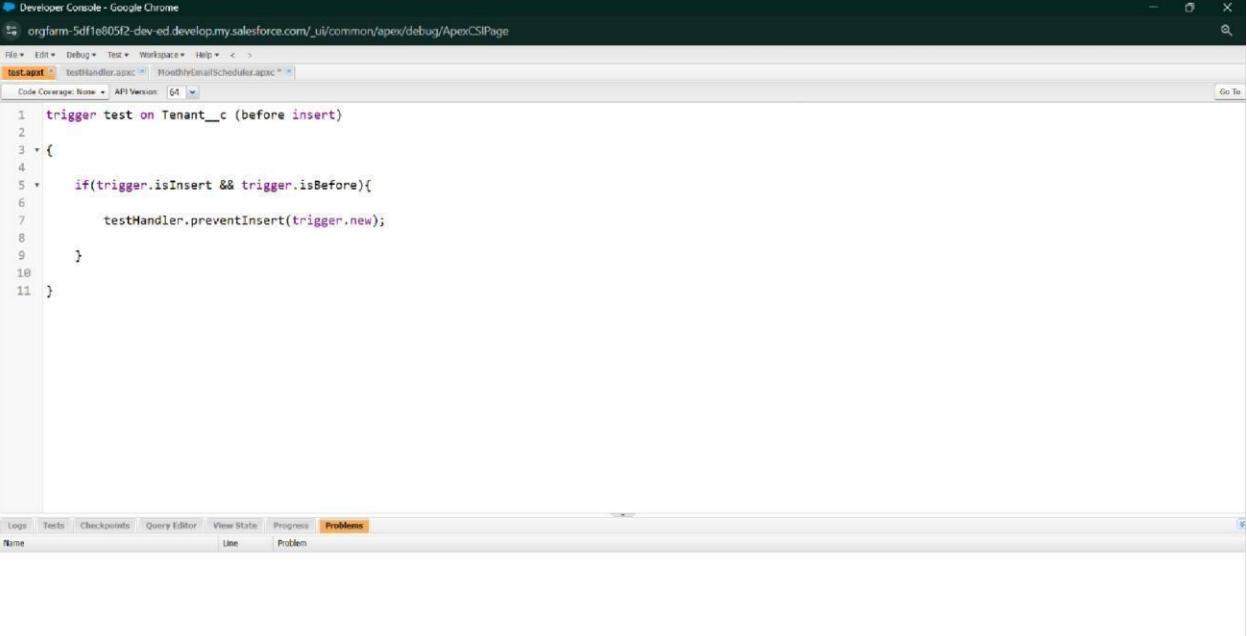
Action	Step Number	Name	Description	Criteria	Assigned Approver	Reject Behavior
Show Actions	Edit	step1			User:Scwmya Team	Final Rejection

## ● Apex Trigger

### Create an Apex Trigger

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

- Code Editor:** File: test.apex, Line: 1, Problem: trigger test on Tenant\_\_c (before insert)
- Search Results:** An open modal window titled "Open" shows a search results table for entity "test". The table has columns: Entity Type, Entity, and Related.
- Bottom Navigation:** Logs, Tests, Checkpoints, Query Editor, View State, Progress, Problems

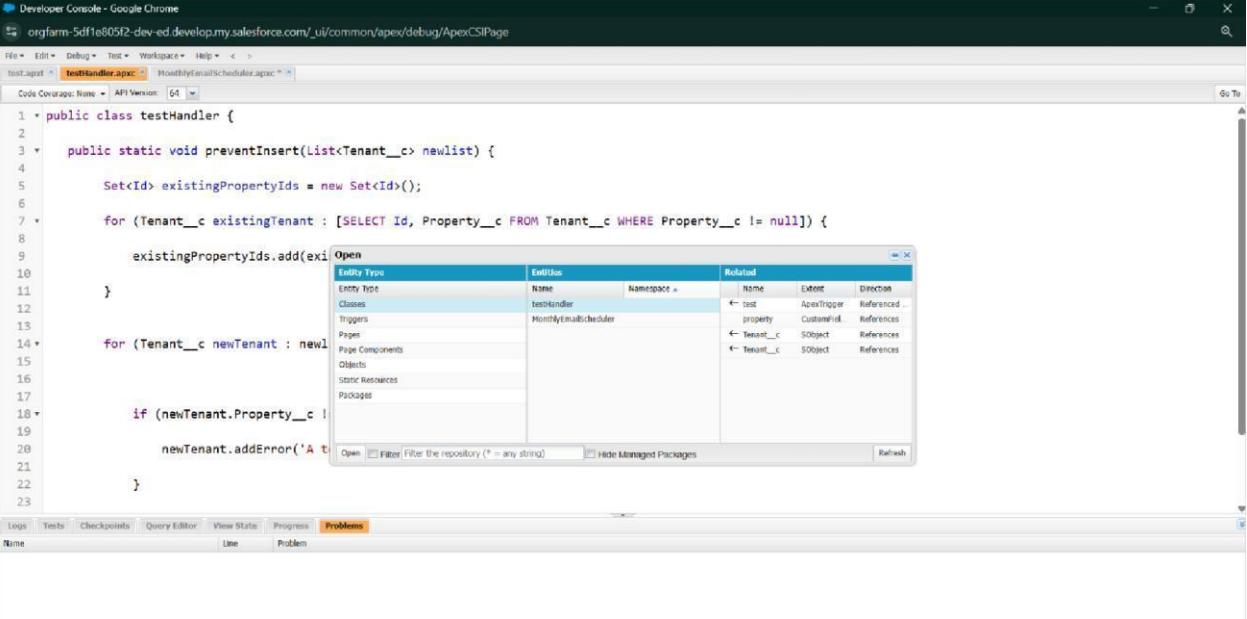


The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is [https://orgfarm-5d1e805f2-dev-ed.develop.my.salesforce.com/\\_ui/common/apex/debug/ApexCSPage](https://orgfarm-5d1e805f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage). The tab bar shows 'test.apex' is active. The code editor contains the following Apex trigger code:

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

The status bar at the bottom indicates 'Logs Tests Checkpoints Query Editor View State Progress Problems'. The 'Problems' tab is selected.

## Create an Apex Handler class



The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is [https://orgfarm-5d1e805f2-dev-ed.develop.my.salesforce.com/\\_ui/common/apex/debug/ApexCSPage](https://orgfarm-5d1e805f2-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSPage). The tab bar shows 'testHandler.apex' is active. The code editor contains the following Apex class code:

```
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 dependency dialog is open over the code editor, listing the relationships between the 'testHandler' class and other entities:

Entity Type	Entity	Relationship
Apex Trigger	test	ApexTrigger
CustomField	property	References
Object	Tenant__c	References
Object	Tenant__c	References

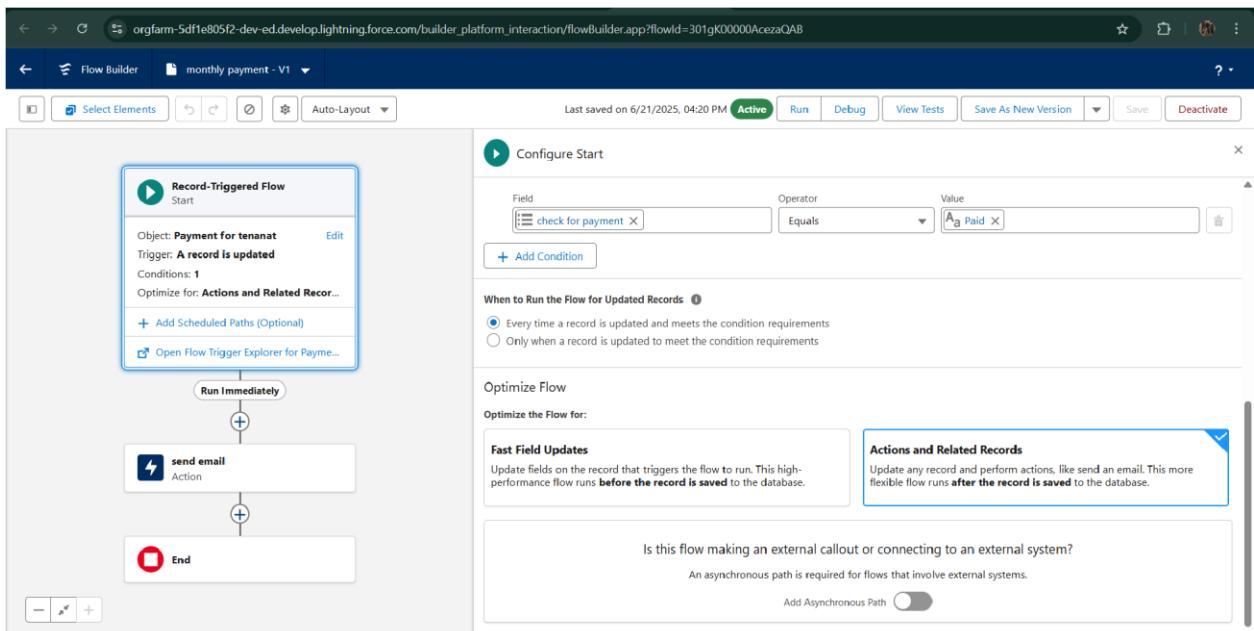
The status bar at the bottom indicates 'Logs Tests Checkpoints Query Editor View State Progress Problems'. The 'Problems' tab is selected.

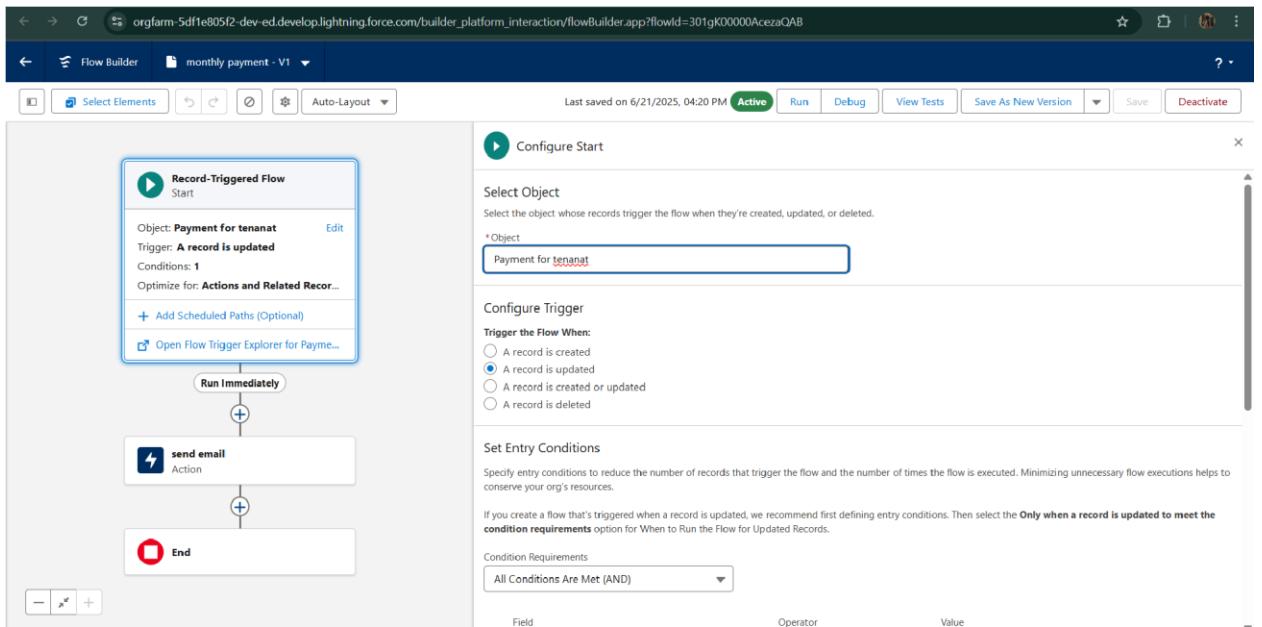
```

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
28 }

```

- FLOWS





- Schedule class:  
Create an Apex Class

```

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

```

Entity Type	Entity	Related
Entity Type	Name Namespace +	
Classes	testHandler	
Triggers	MonthlyEmailScheduler	Email CustomField... References...
Pages		← Tenant__c Subject References...
Page Components		← Tenant__c Subject References...
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             String emailSubject = 'Monthly Rent Payment Due';
26
27             Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
28
29             email.setToRecipients(new String[]{recipientEmail});
30
31             email.setSubject(emailSubject);
32
33             email.setPlainTextBody(emailContent);
34
35             messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
36
37         }
38
39     }
40
41 }

```

## Schedule Apex class

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

- Apex Class:** MonthlyEmailScheduler
- Name:** MonthlyEmailScheduler
- Namespace Prefix:** None
- Created By:** Sowmya\_Team . 6/23/2025, 2:46 AM
- Status:** Active
- Code Coverage:** 0% (0/15)
- Last Modified By:** Sowmya\_Team . 6/23/2025, 2:47 AM

The Class Body tab displays the Apex code for the MonthlyEmailScheduler class:

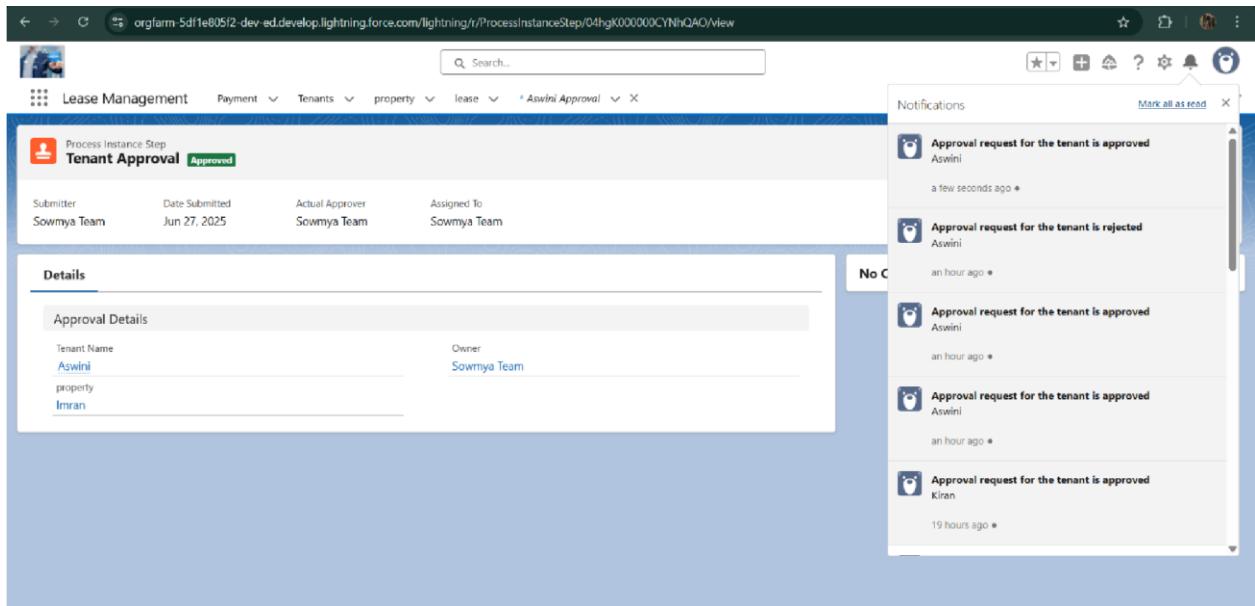
```

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             String emailSubject = 'Monthly Rent Payment Due';
26
27             Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
28
29             email.setToRecipients(new String[]{recipientEmail});
30
31             email.setSubject(emailSubject);
32
33             email.setPlainTextBody(emailContent);
34
35             messaging.sendEmail(new Messaging.SingleEmailMessage[]{email});
36
37         }
38
39     }
40
41 }

```

The screenshot shows a Salesforce Lightning page for a tenant named 'Aswini'. The 'Details' tab is selected. The tenant's name is 'Aswini', email is 'aswiniamaraadi15@gmail.com', phone is '(905) 223-5567', and status is 'Leaving'. The property is listed as 'Imran'. The record was created by 'Sowmya Team' on 6/26/2025, 6:05 AM, and last modified by 'Sowmya Team' on 6/26/2025, 11:06 PM. A sidebar on the right shows the 'Activity' section with a message: 'No activities to show. Get started by sending an email, scheduling a task, and more.' A context menu is open, showing options like 'New Case', 'New Lead', 'Delete', 'Clone', 'Change Owner', 'Printable View', 'Submit for Approval', and 'Edit Labels'. The URL in the browser is [https://orgfarm-5d1e805f2-dev-ed.develop.lightning.force.com/lightning/r/Tenant\\_\\_c/a01gK0000BAhdGQAT/view](https://orgfarm-5d1e805f2-dev-ed.develop.lightning.force.com/lightning/r/Tenant__c/a01gK0000BAhdGQAT/view).

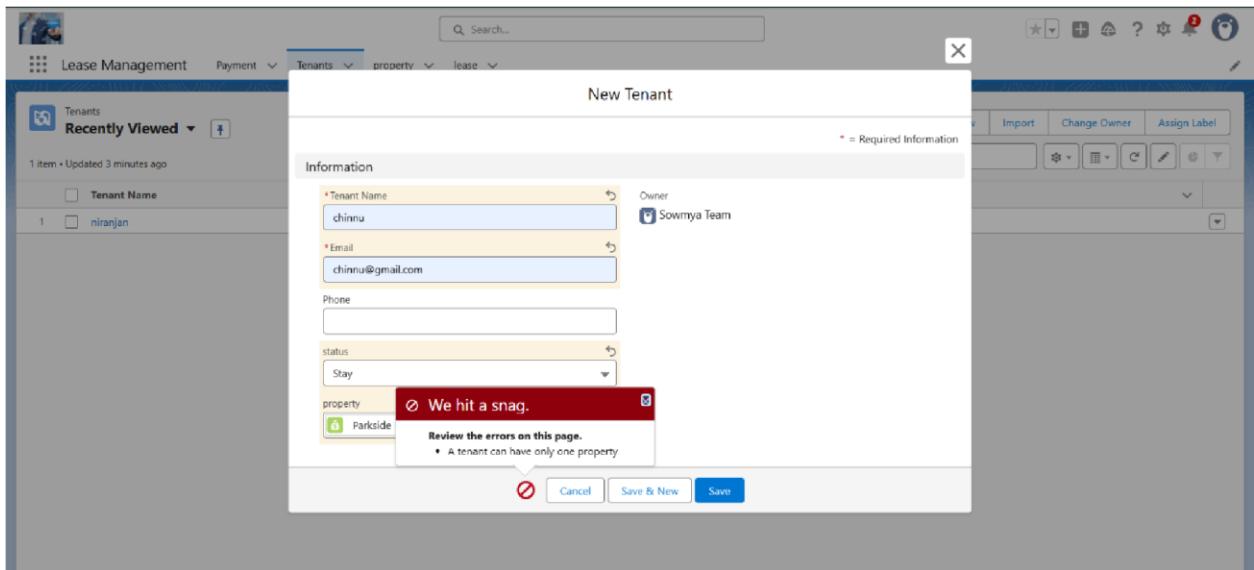
The screenshot shows the same Salesforce Lightning page for 'Aswini'. A green success message at the top states 'Tenant was submitted for approval.' The rest of the page is identical to the first screenshot, showing the tenant details and the activity sidebar. The URL in the browser is [https://orgfarm-5d1e805f2-dev-ed.develop.lightning.force.com/lightning/r/Tenant\\_\\_c/a01gK0000BAhdGQAT/view](https://orgfarm-5d1e805f2-dev-ed.develop.lightning.force.com/lightning/r/Tenant__c/a01gK0000BAhdGQAT/view). The system tray at the bottom shows a weather icon (30°C Cloudy), a search bar, and various application icons.



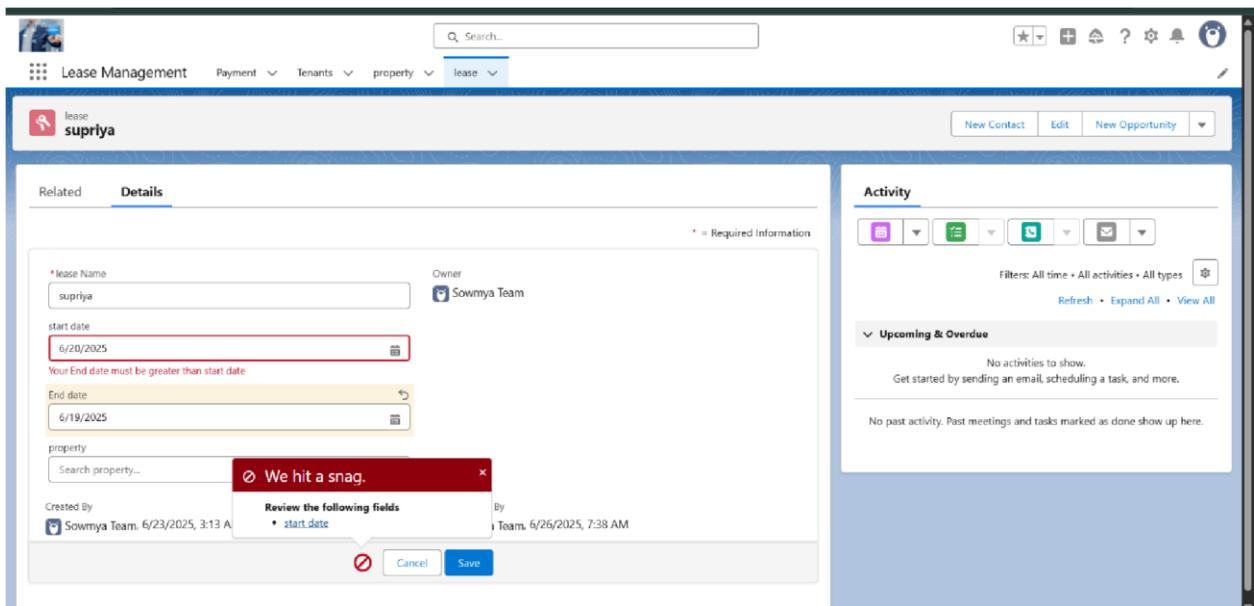
# FUNCTIONAL AND PERFORMANCE TESTING

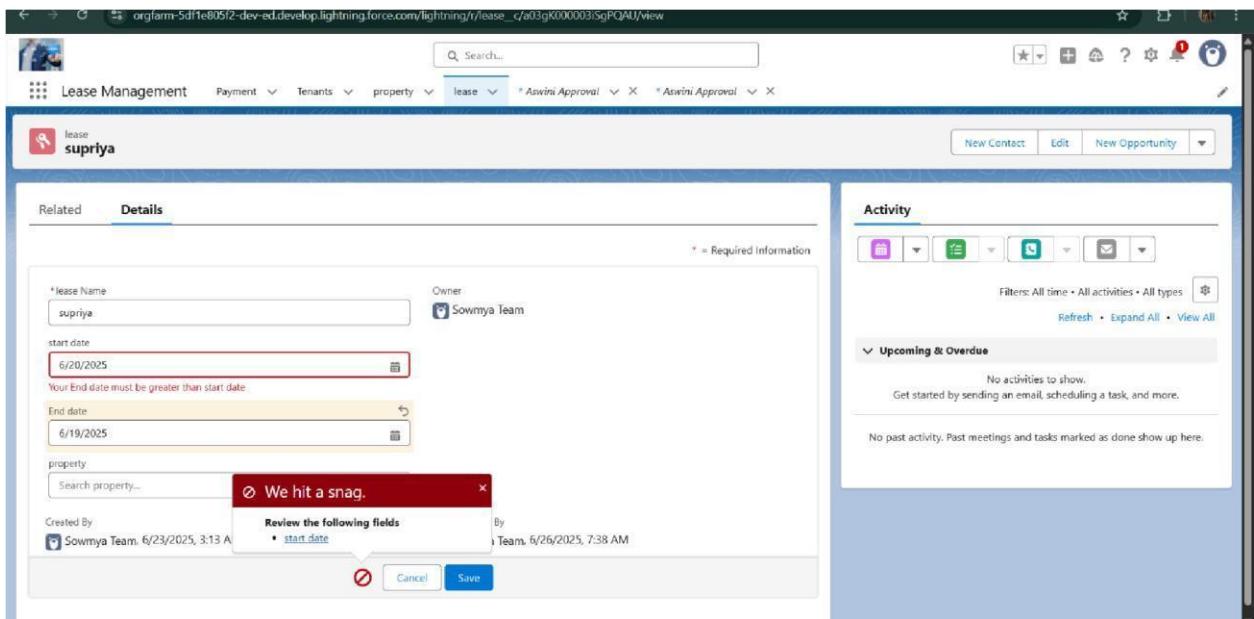
## Performance Testing

- Trigger validation by entering duplicate tenant-property records

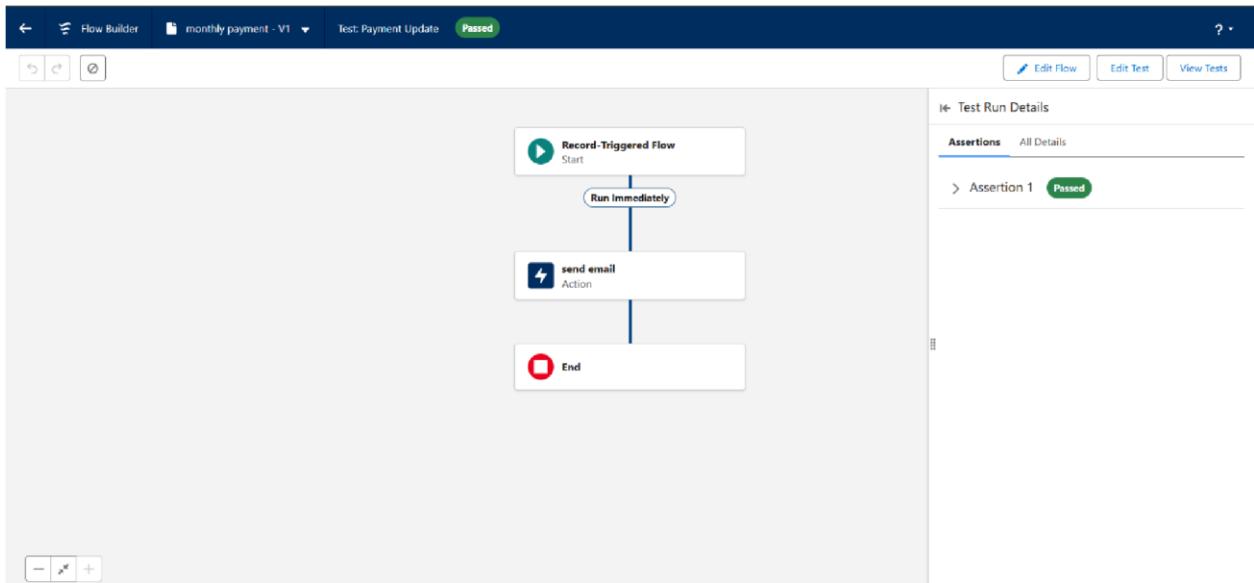


- Validation Rule checking





- Test flows on payment update



- Approval process validated through email alerts and status updates

**Tenant niranjan**

**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

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

**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

**Payment (2)**

Payment Name	New
Jack	
Rahul	

# RESULTS

## Output Screenshots

- Tabs for Property, Tenant, Lease, Payment

**Custom Tabs**

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.

Action	Label	Tab Style	Description
Edit   Del	Lease	Keys	
Edit   Del	Payment	Credit Card	
Edit   Del	property	Sack	
Edit   Del	Tenants	Map	

**Web Tabs**

No Web Tabs have been defined.

**Visualforce Tabs**

No Visualforce Tabs have been defined.

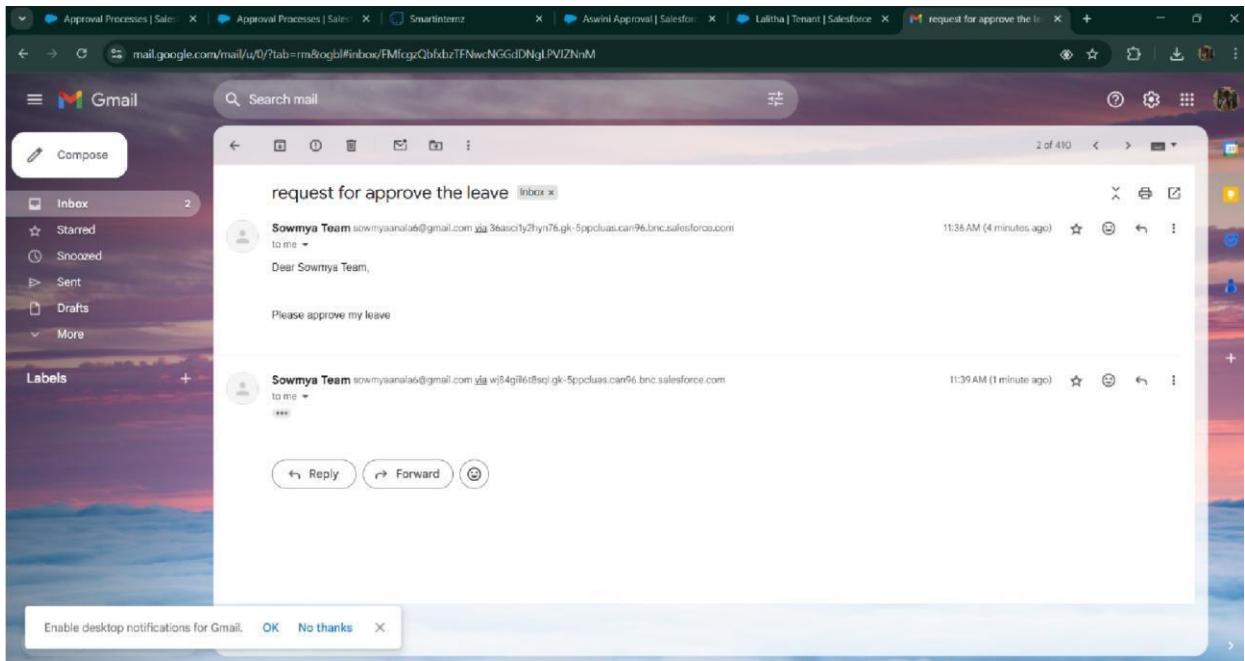
- Email alerts

**Approval History**

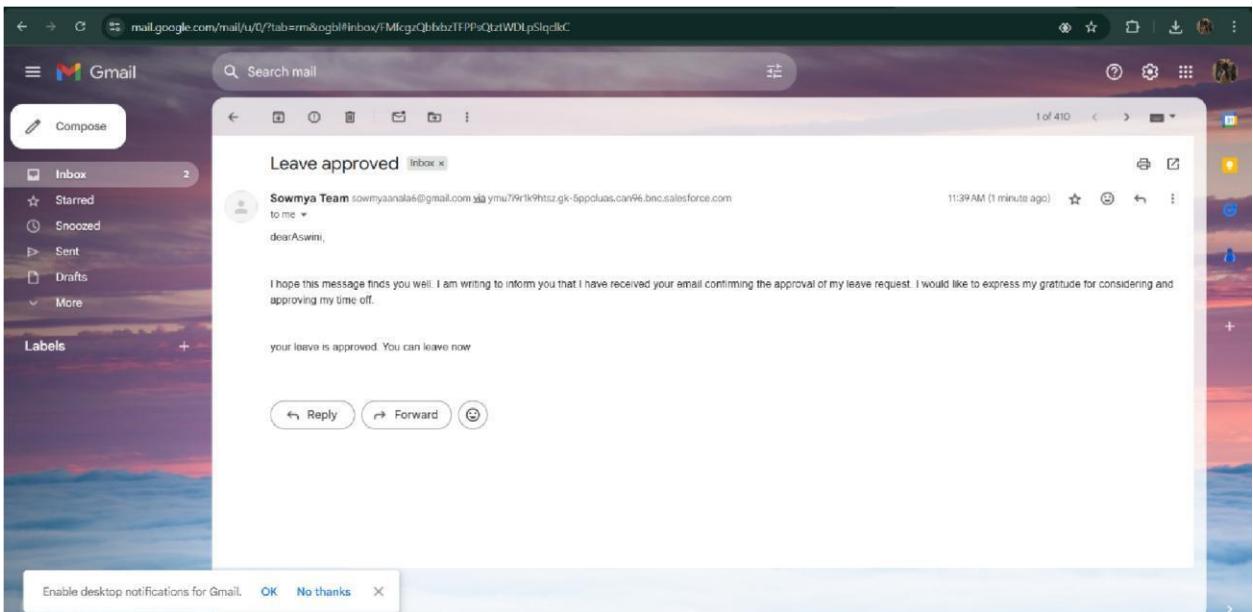
8 items • Sorted by Is Pending • Updated a few seconds ago

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

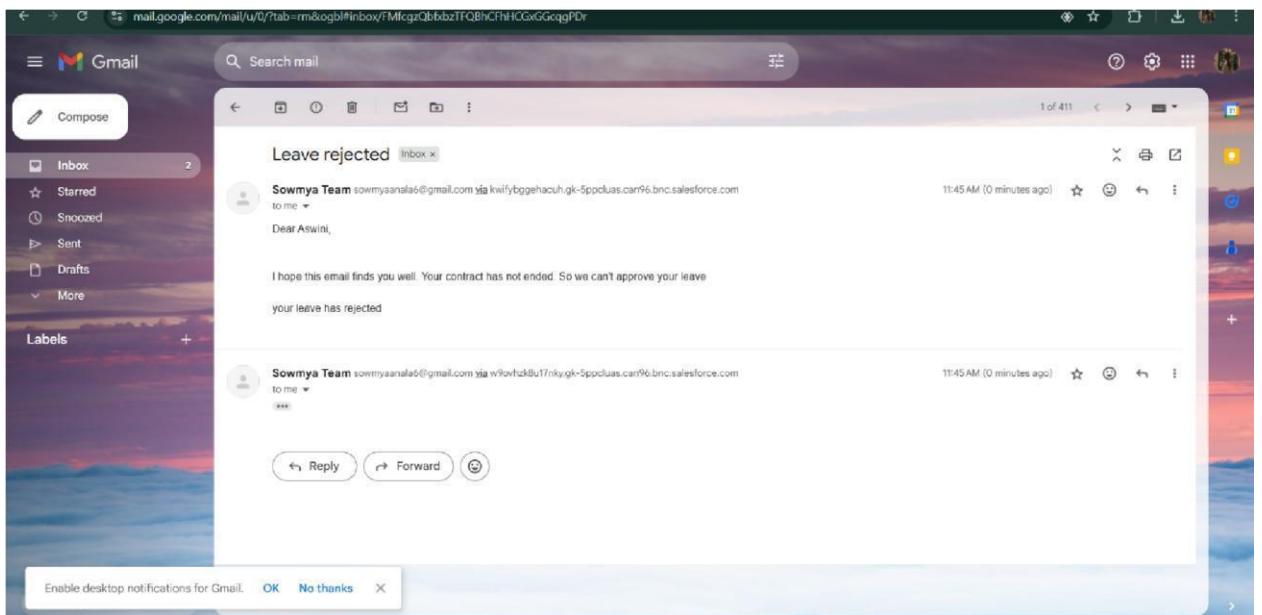
- Request for approve the leave



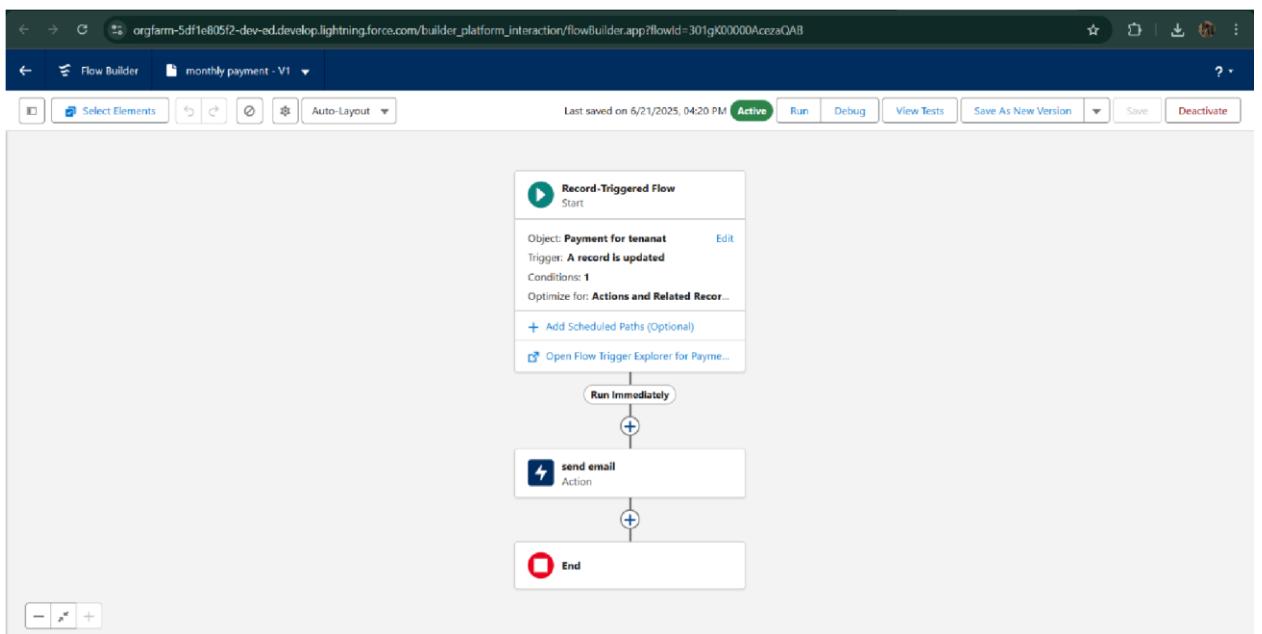
- Leave approved



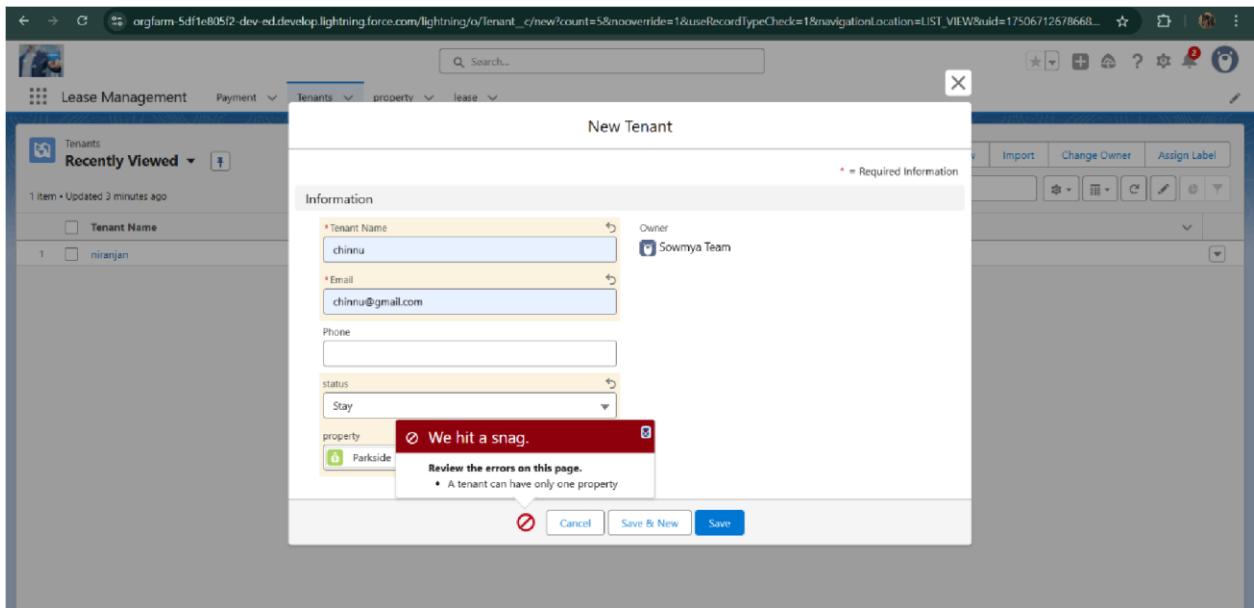
- Leave rejected



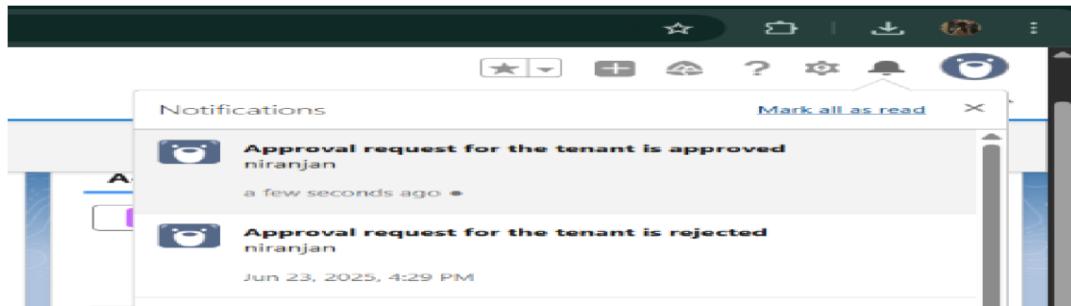
- Flow runs



- Trigger error messages



- Approval process notifications



## ADVANTAGES & DISADVANTAGES

### ADVANTAGE

- Lower upfront cost
- Increased flexibility
- Reduced responsibility

### DISADVANTAGE

- Higher Long-term costs
- No ownership or equity
- Lack of control

# 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});  
  
}  
  
}  
  
}
```