SALESFORCE DEVELOPMENT

PROJECT REPORT

Submitted by

DHARSHINI S R	710021104007
SWETHA B	710021104031
KAVIARASU S	710021104311
JAYSURYA R K	710021104705

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

ANNA UNIVERSITY REGIONAL CAMPUS , COIMBATORE

ANNA UNIVERSITY : COIMBATORE – 641046

LEASE MANAGEMENT

1. Project Overview

The Lease Management System is designed to streamline and automate the process of managing lease agreements for property management companies, landlords, and tenants. This system aims to improve efficiency by digitizing lease documentation, rental payment tracking, lease renewals, and tenant information management.

The application provides a centralized platform to handle multiple leases across various properties, allowing stakeholders to monitor lease statuses, upcoming renewals, payment schedules, and tenant communications.

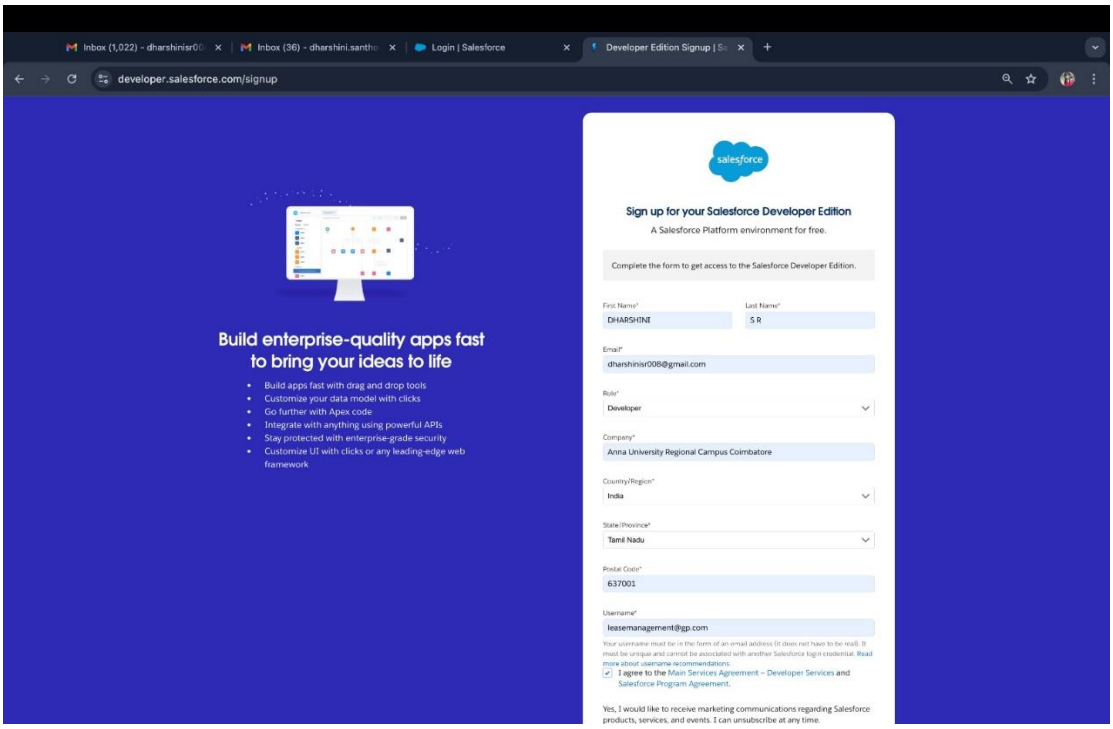
2. Short Description

A Lease Management System is a digital solution designed to streamline the administration of property leases by centralizing lease agreements, tracking rental payments, and managing tenant information. It automates reminders for payments and lease renewals, provides a tenant portal for easy communication, and supports financial reporting to improve operational efficiency for property managers and landlords. The system reduces paperwork, ensures timely compliance, and enhances transparency for all stakeholders involved in property leasing.

3. Developer Account Creation

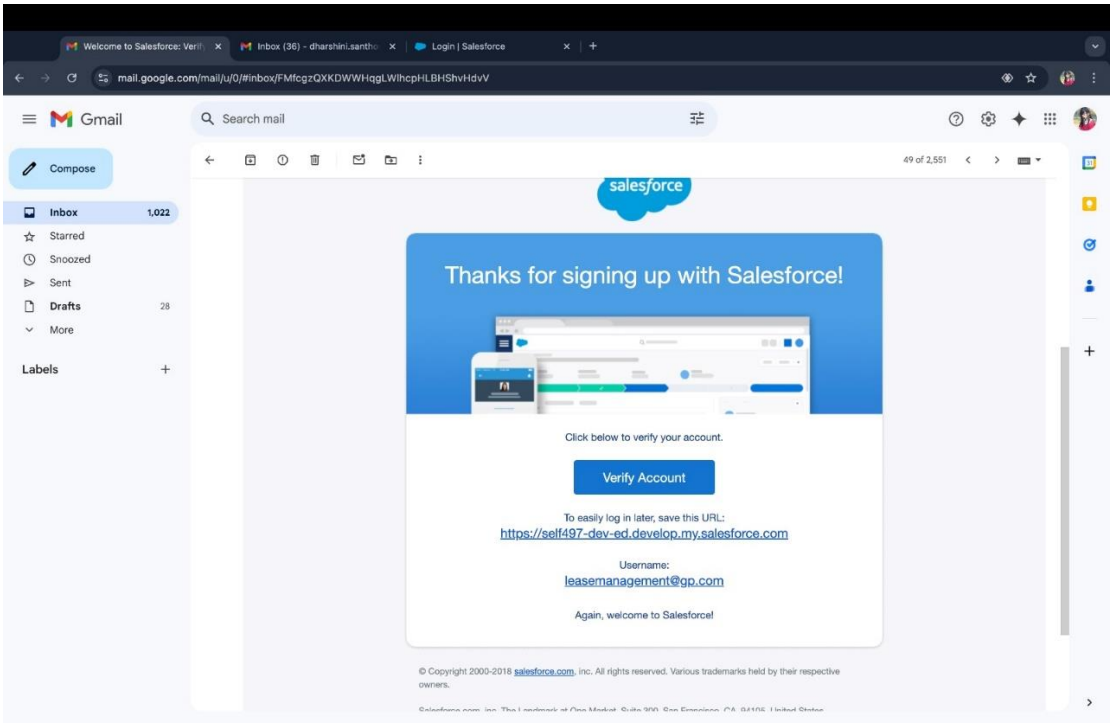
1. Sign-Up Process

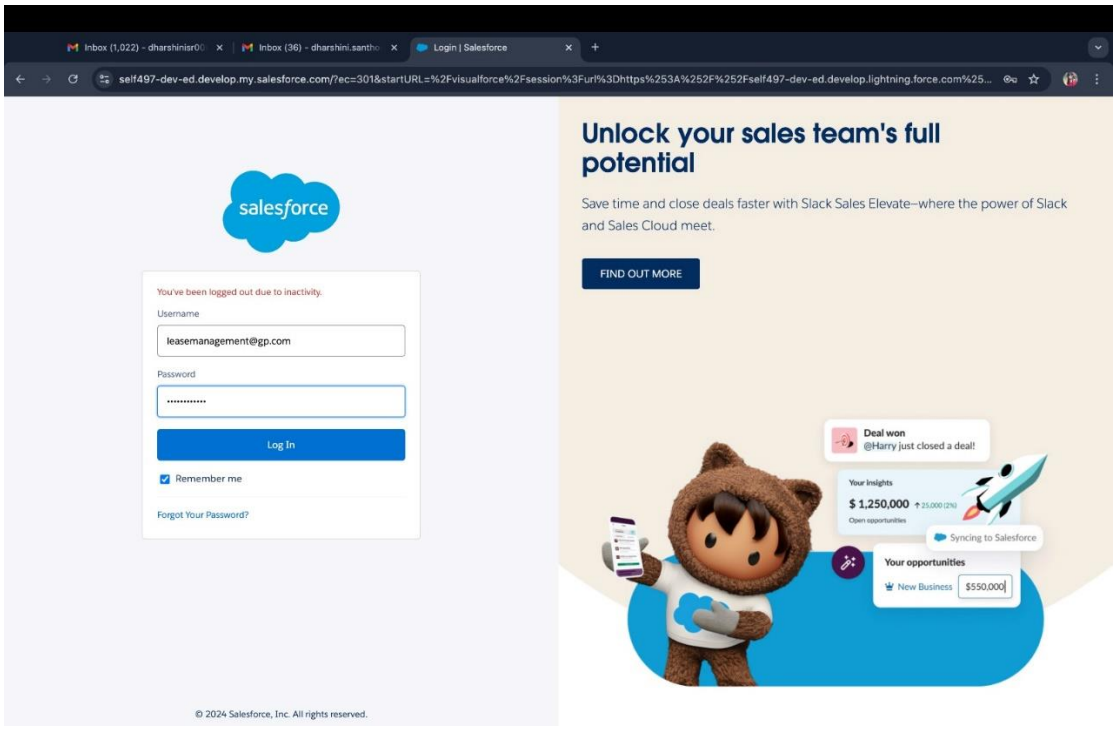
- Go to Salesforce Developer Sign-Up.
- Enter your **First and Last Name**, **Email**, and set **Role** as "Developer."
- Input your **Company** (College Name), **Country** (India), **Postal Code**, and **Username** (formatted as username@organization.com).
- Click **Sign Me Up** after filling out the form.



2. Account Activation

- Open the inbox of the email used for registration, locate the Salesforce verification email, and click **Verify Account**.
- Set a password, choose a security question, and log into your Salesforce account to access the setup page.





4. Objects in Salesforce

Go to Setup:

- Log in to Salesforce, click on the **Setup** icon (gear icon), and select **Setup**.

Access Object Manager:

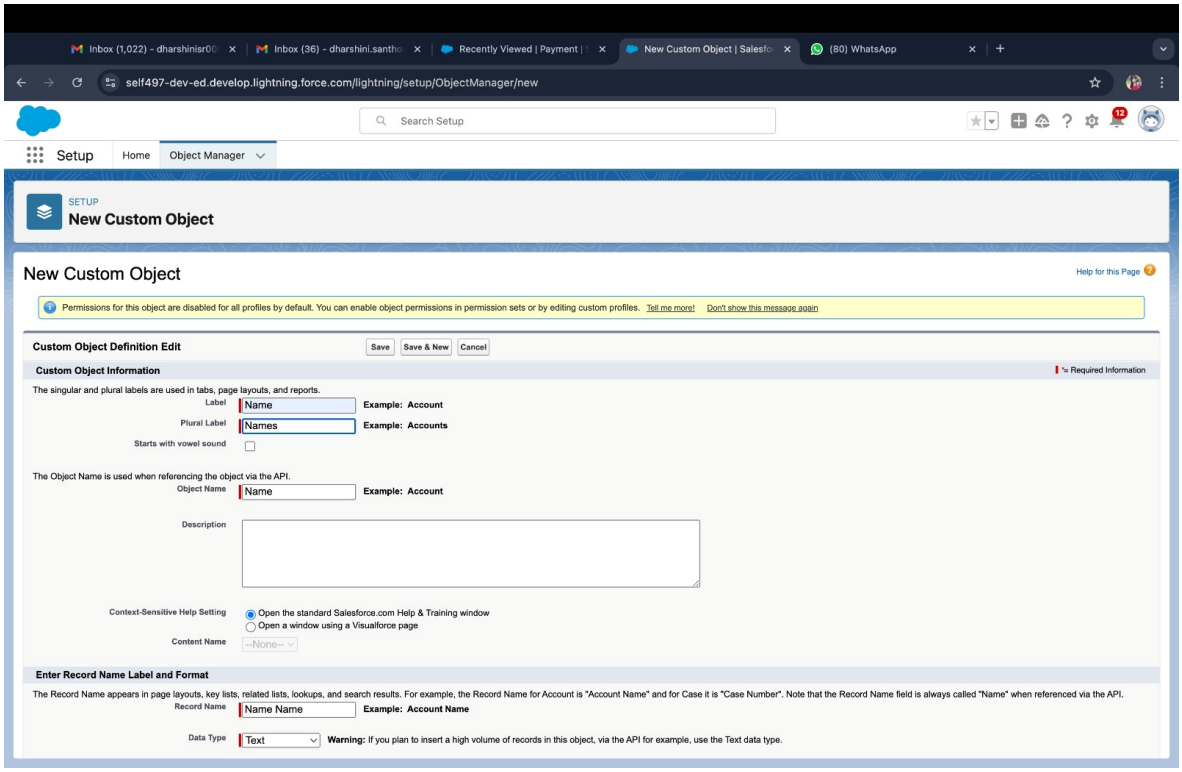
- In the Setup page, find and open **Object Manager**.

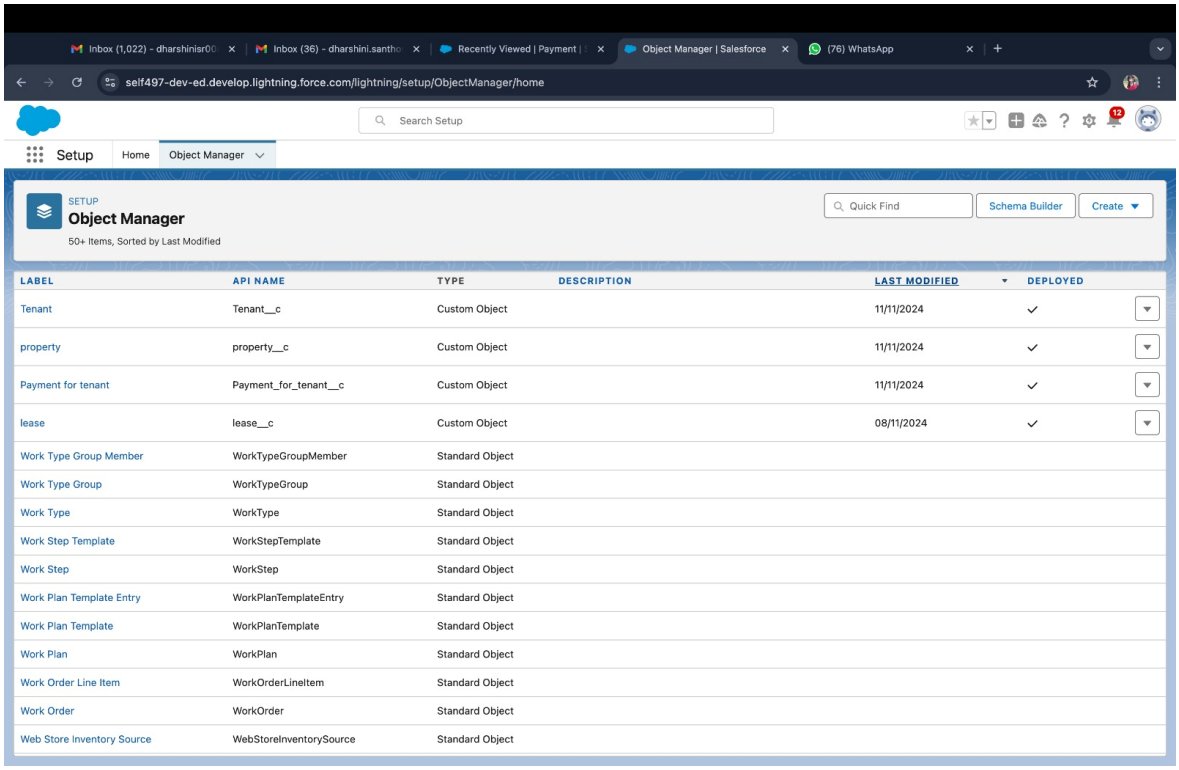
Create Each Object:

- **Click on Create** > **Custom Object**.
- **Enter the Label and Plural Label**:
- **Property**: Label = "Property", Plural Label = "Properties"
- **Tenant**: Label = "Tenant", Plural Label = "Tenants"
- **Payment for Tenant**: Label = "Payment for Tenant", Plural Label = "Payments"
- **Lease**: Label = "Lease", Plural Label = "Leases"
- **Set Record Name**:
- Enter "Object Name + Name" (e.g., **Property Name** for Property).
- Choose **Data Type: Text**.
- **Enable Options**:
- Check **Allow Reports**, **Track Field History**, **Allow Activities**, and **Allow Search**.

- **Click Save** to create the object.

Repeat Steps for each object as listed above (Property, Tenant, Payment for Tenant, Lease).



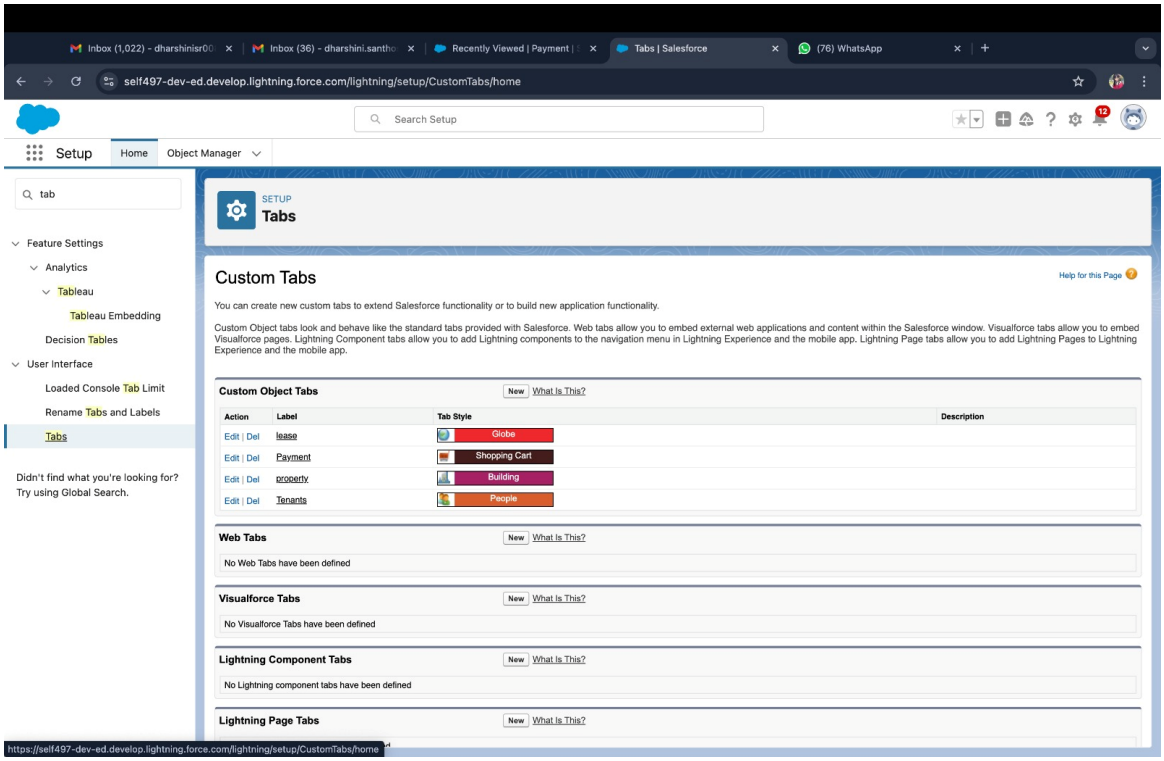


5. Creation Of Tabs

1. **Go to Setup**:
   - Log in to Salesforce, click the **Setup** icon (gear icon), and go to **Setup**.
2. **Find Tabs**:
   - In the **Quick Find** bar on the left, type **Tabs**.
   - Click on **Tabs**.
3. **Create a New Custom Object Tab**:
   - Under **Custom Object Tabs**, click **New**.
4. **Select Object**:
   - In the **Select Object** dropdown, choose the relevant object (e.g., **Property**, **Tenant**, **Payment for Tenant**, or **Lease**).
5. **Select Tab Style**:
   - Choose a **Tab Style** that best represents the object (e.g., a home icon for Property, a payment icon for Payment, etc.).
6. **Set Profile Visibility**:
   - Click **Next** to move to the Profiles page.
   - Leave profile visibility settings as **default**.
7. **Configure Custom App Visibility**:
   - On the next page, under **Add to Custom Apps**, uncheck **Include Tab** if you don't want it in any specific app initially.
8. **Verify Personalization Options**:
   - Ensure that the **Append tab to users' existing personal customizations** option is checked, allowing users to add the tab to their own setup.
9. **Save**:
   - Click **Save** to create the tab.

10. **Repeat Steps** for each object:

- Follow the above steps for each object: **Property**, **Tenant**, **Payment for Tenant**, and **Lease**.



6. Creation Of Fields

- **Go to Setup**:

  Log in to Salesforce, click the **Setup** icon (gear icon), and go to **Setup**.

- **Find Tabs**:

  ✓ In the **Quick Find** bar on the left, type **Tabs**.
  ✓ Click on **Tabs**.

- **Create a New Custom Object Tab**:

  ✓ Under **Custom Object Tabs**, click **New**.

- **Select Object**:

  ✓ In the **Select Object** dropdown, choose the relevant object (e.g., **Property**, **Tenant**, **Payment for Tenant**, or **Lease**).

- **Select Tab Style**:

- ✓ Choose a **Tab Style** that best represents the object (e.g., a home icon for Property, a payment icon for Payment, etc.).

- **Set Profile Visibility**:

  - ✓ Click **Next** to move to the Profiles page.
  - ✓ Leave profile visibility settings as **default**.

- **Configure Custom App Visibility**:

  - ✓ On the next page, under **Add to Custom Apps**, uncheck **Include Tab** if you don't want it in any specific app initially.

- **Verify Personalization Options**:

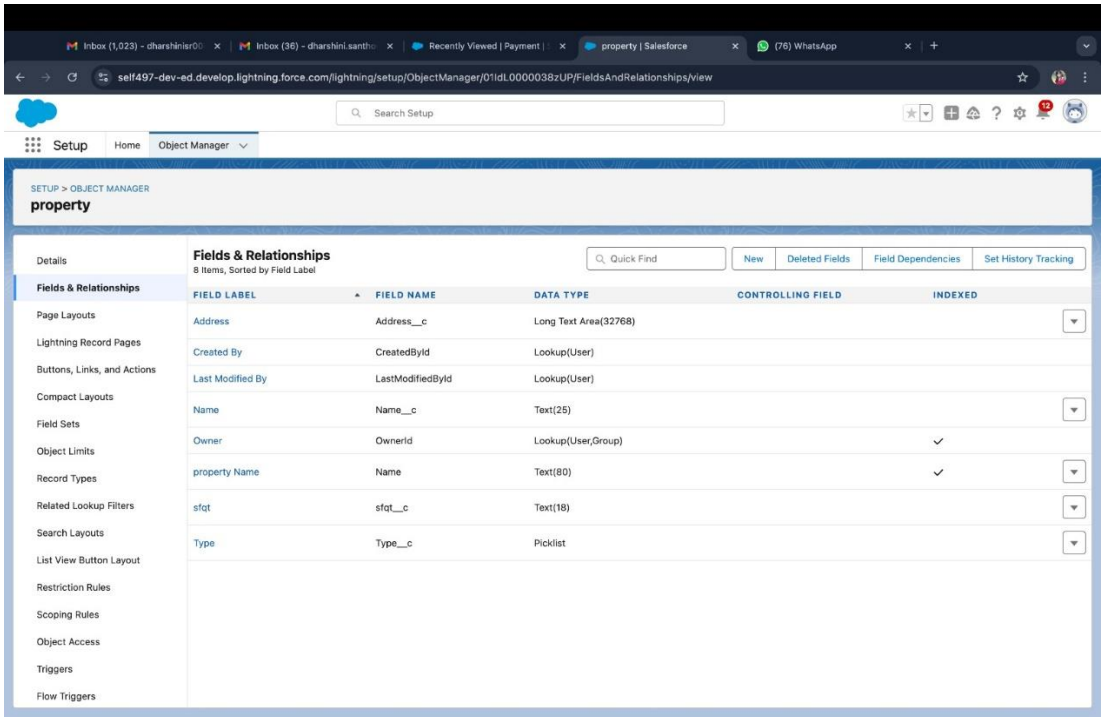
  - ✓ Ensure that the **Append tab to users' existing personal customizations** option is checked, allowing users to add the tab to their own setup.

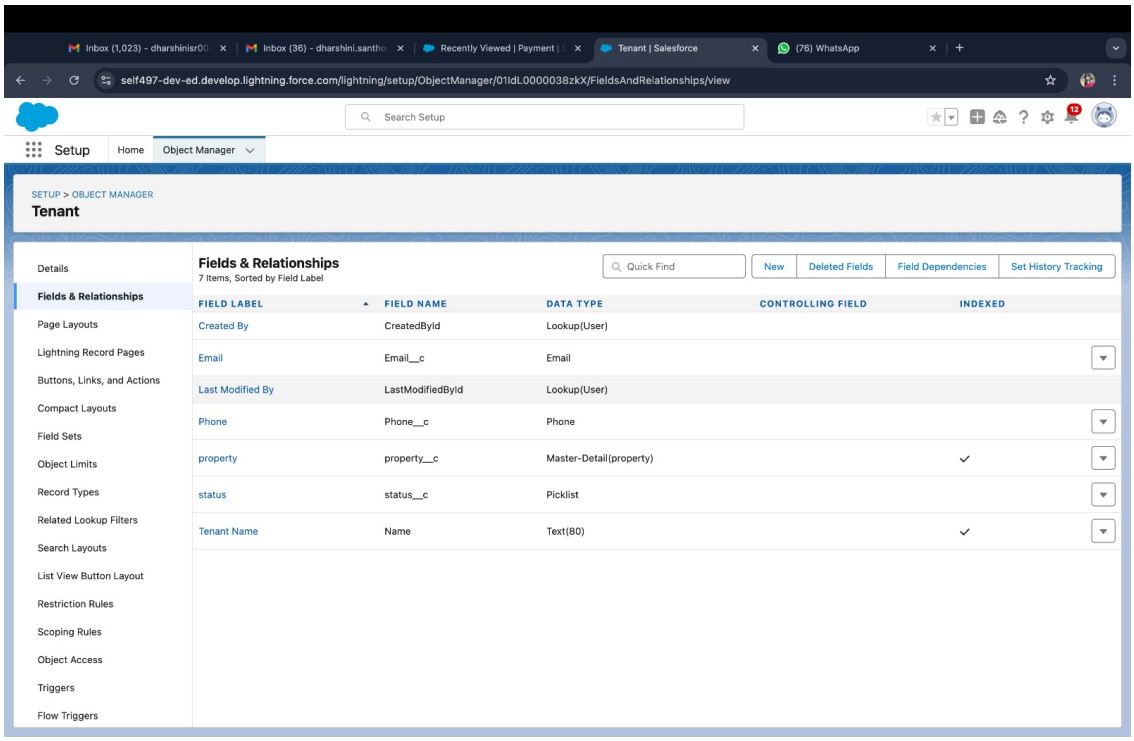
- **Save:**

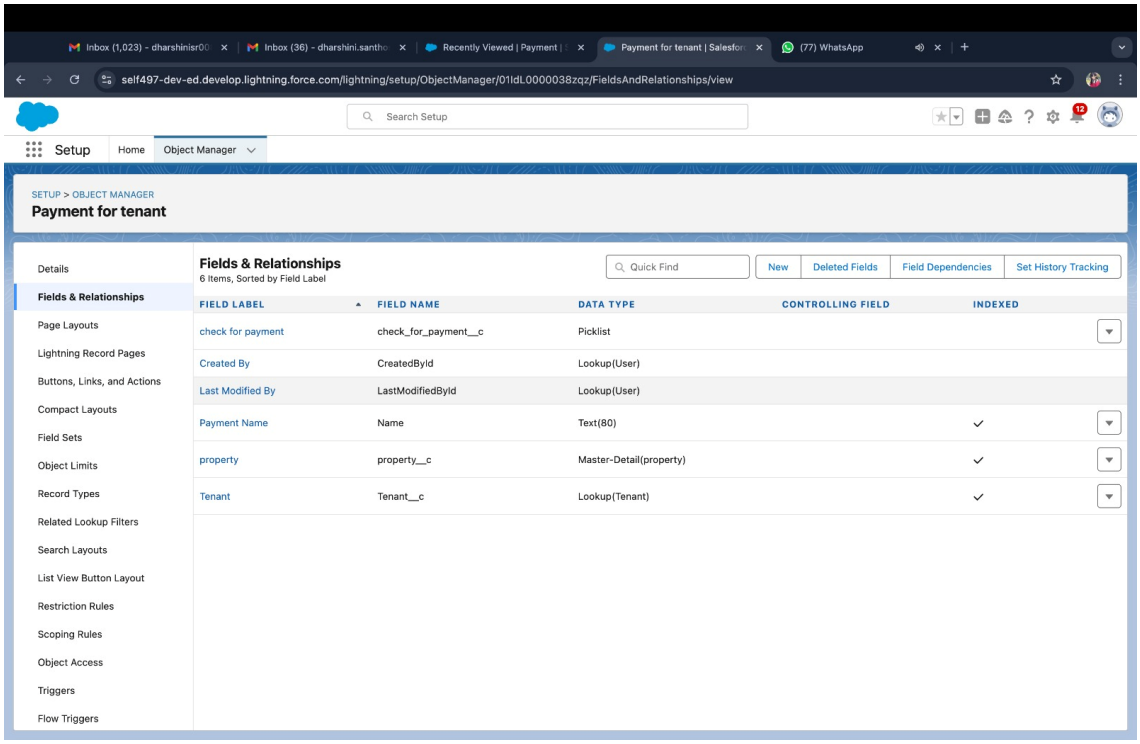
  - ✓ Click **Save** to create the tab.

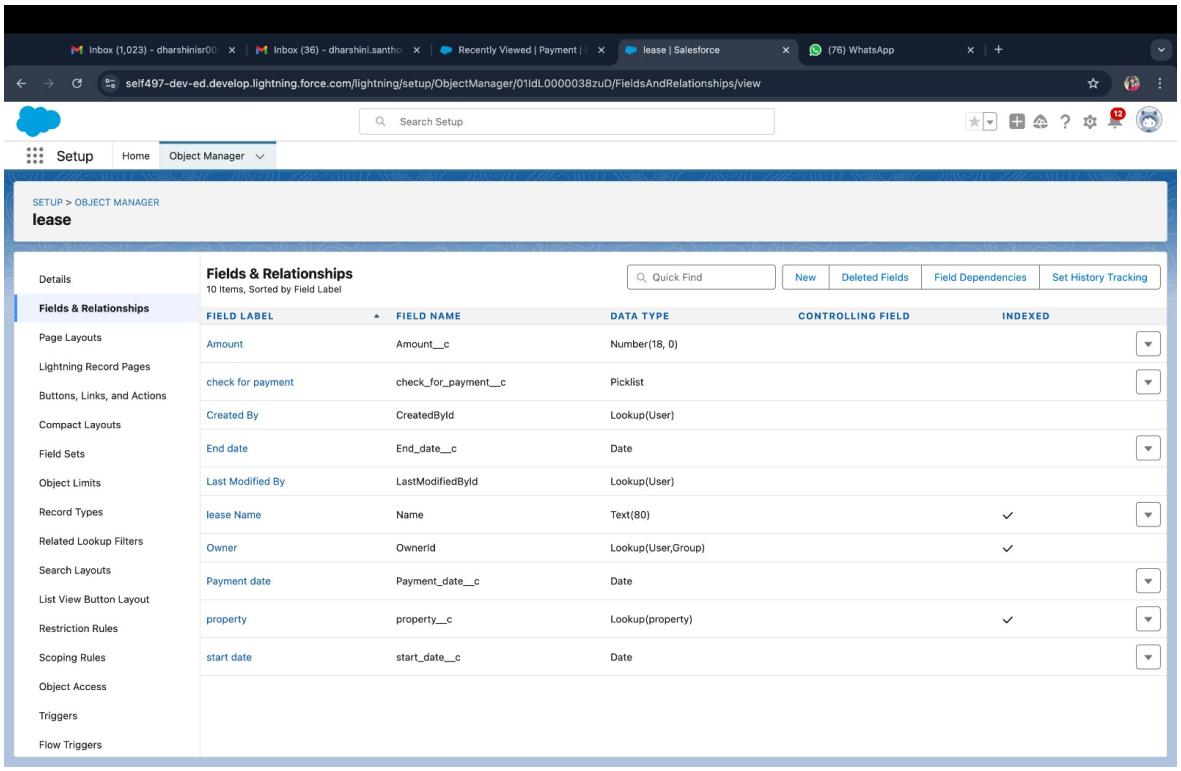
- **Repeat Steps** for each object:

  - ✓ Follow the above steps for each object: **Property**, **Tenant**, **Payment for Tenant**, and **Lease**.

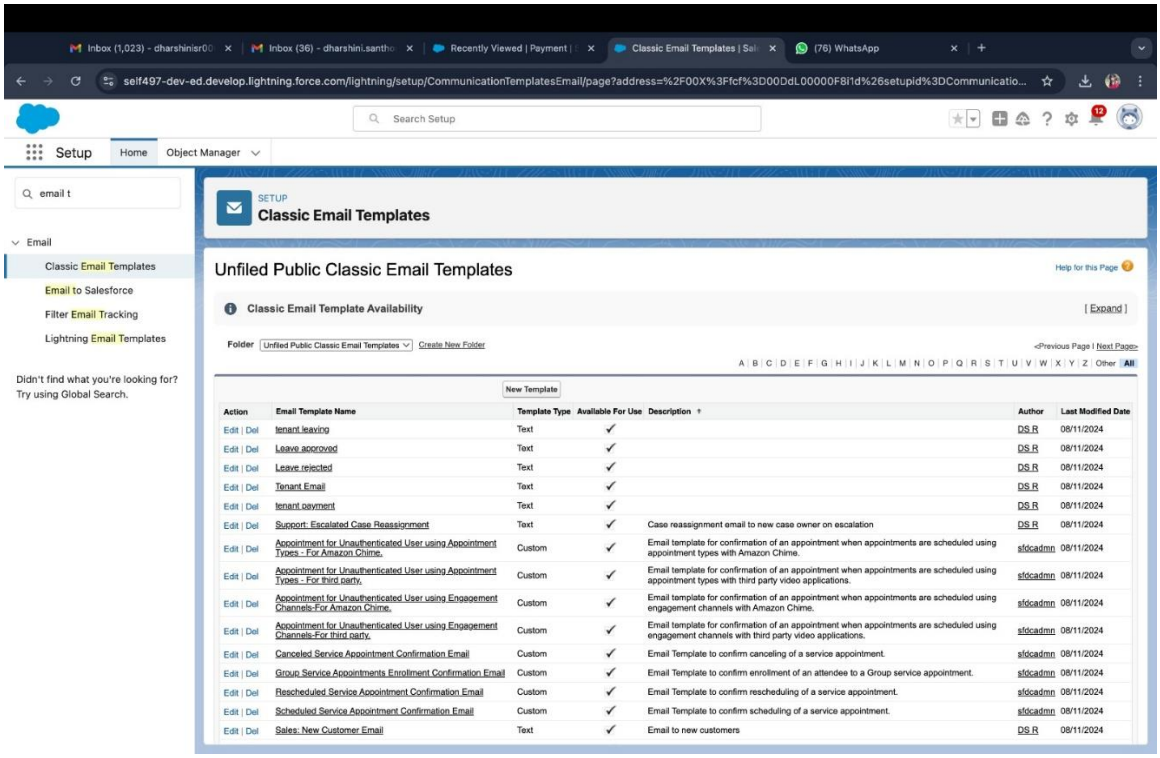



Inbox (1,023) - dharshinisr00
Inbox (36) - dharshini.santho
Recently Viewed | Payment |
Tenant | Salesforce
(76) WhatsApp
self497-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01IdL0000038zkX/FieldsAndRelationships/view
Search Setup
Setup
Home
Object Manager
SETUP > OBJECT MANAGER
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
Object Access
Triggers
Flow Triggers
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)
Email
Email__c
Email
Last Modified By
LastModifiedById
Lookup(User)
Phone
Phone__c
Phone
property
property__c
Master-Detail(property)
status
status__c
Picklist
Tenant Name
Name
Text(80)


Inbox (1,023) - dharshinisr00
Inbox (36) - dharshini.santho
Recently Viewed | Payment |
Payment for tenant | Salesfor
(77) WhatsApp
self497-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01IdL0000038zqz/FieldsAndRelationships/view
Search Setup
Setup
Home
Object Manager
SETUP > OBJECT MANAGER
Payment for 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
Object Access
Triggers
Flow Triggers
Fields & Relationships
6 Items, Sorted by Field Label
Quick Find
New
Deleted Fields
Field Dependencies
Set History Tracking
FIELD LABEL
FIELD NAME
DATA TYPE
CONTROLLING FIELD
INDEXED
check for payment
check_for_payment__c
Picklist
Created By
CreatedById
Lookup(User)
Last Modified By
LastModifiedById
Lookup(User)
Payment Name
Name
Text(80)
property
property__c
Master-Detail(property)
Tenant
Tenant__c
Lookup(Tenant)



7. Email Template



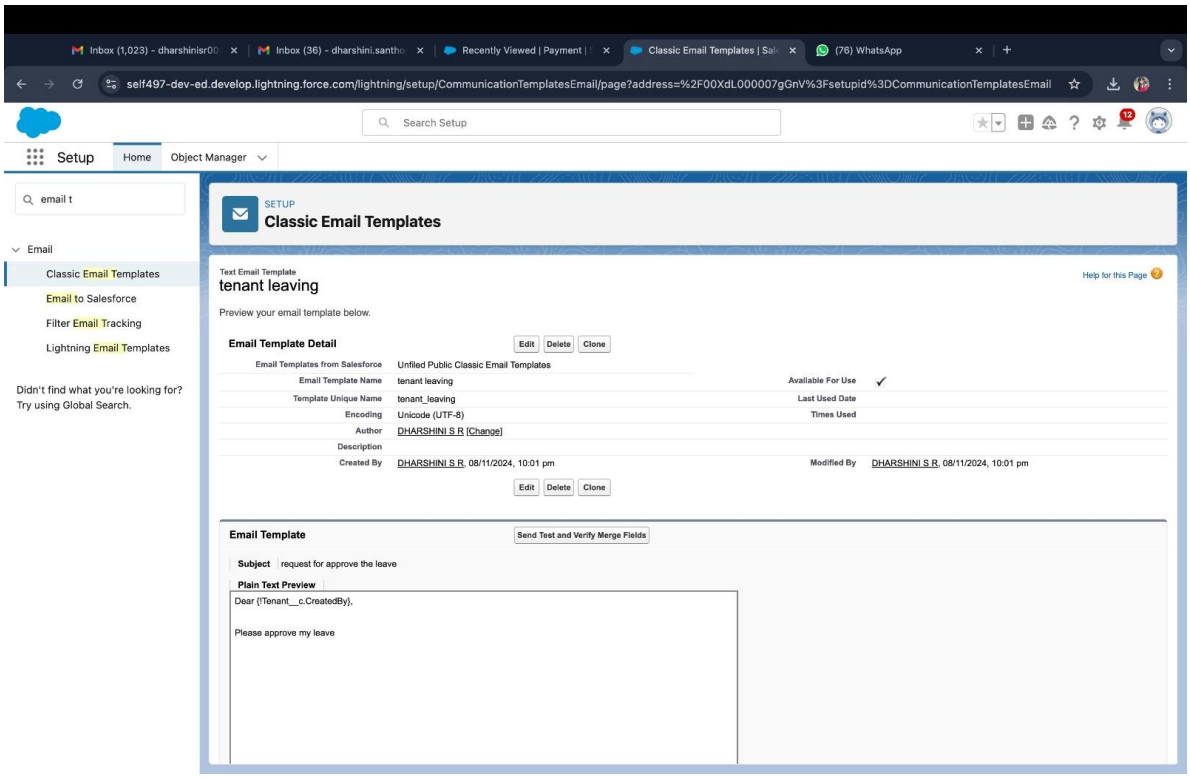
Email Template for Tenant Leaving Request

1. Go to Setup > enter "Email Template" in the Quick Find box > select Classic Email Template.
2. Click New Email Template > Choose Text format.
   - Folder: Unfiled Public Classic Email Templates
   - Check Available for Use.
3. Template Name: Tenant Leaving
   - Unique Name: Auto-populated
4. Subject: Request for approval to leave
5. Email Body:

   Dear {!Tenant__c.CreatedBy},

   Please approve my leave request.

6. Click Save.



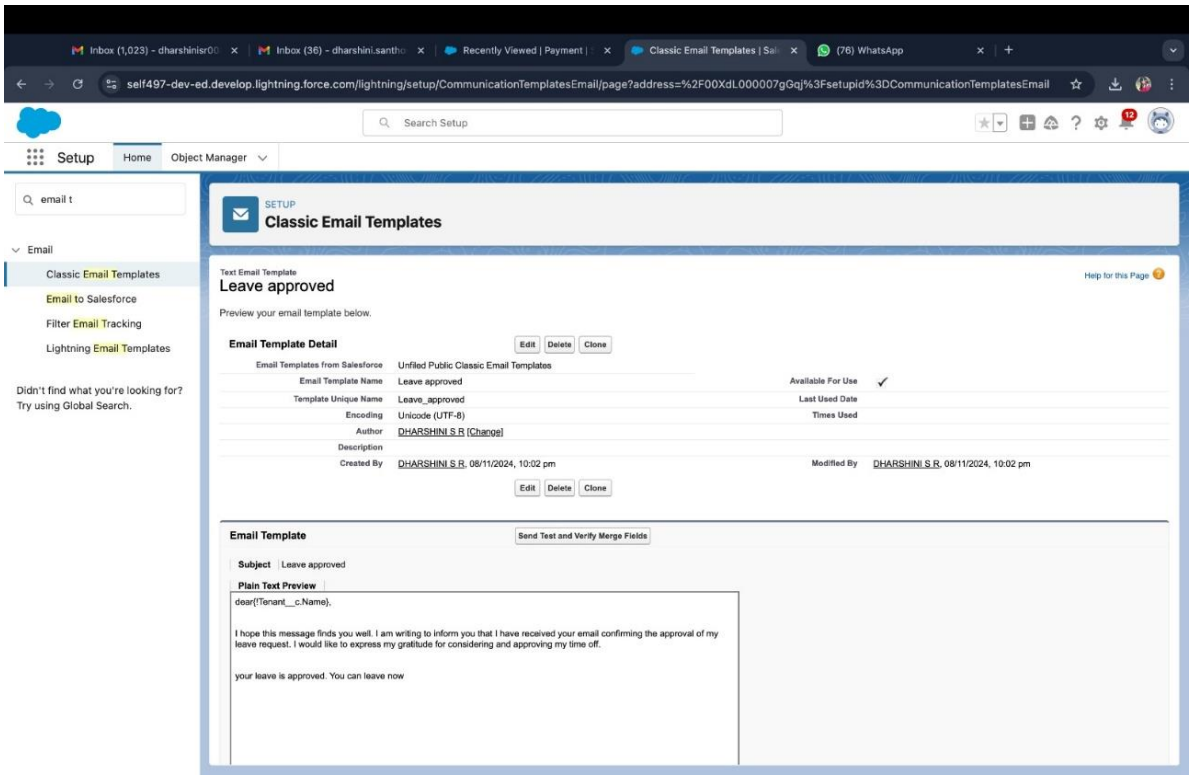
Email Template for Leave Approved

1. Go to Setup > enter "Email Template" in the Quick Find box > select Classic Email Template.
2. Click New Email Template > Choose Text format.
   - Folder: Unfiled Public Classic Email Templates
   - Check Available for Use.
3. Template Name: Leave Approved
   - Unique Name: Auto-populated
4. Subject: Leave Approved
5. Email Body:

   Dear {!Tenant__c.Name},

   I hope this message finds you well. Your leave has been approved. You may proceed with your departure.

6. Click Save.



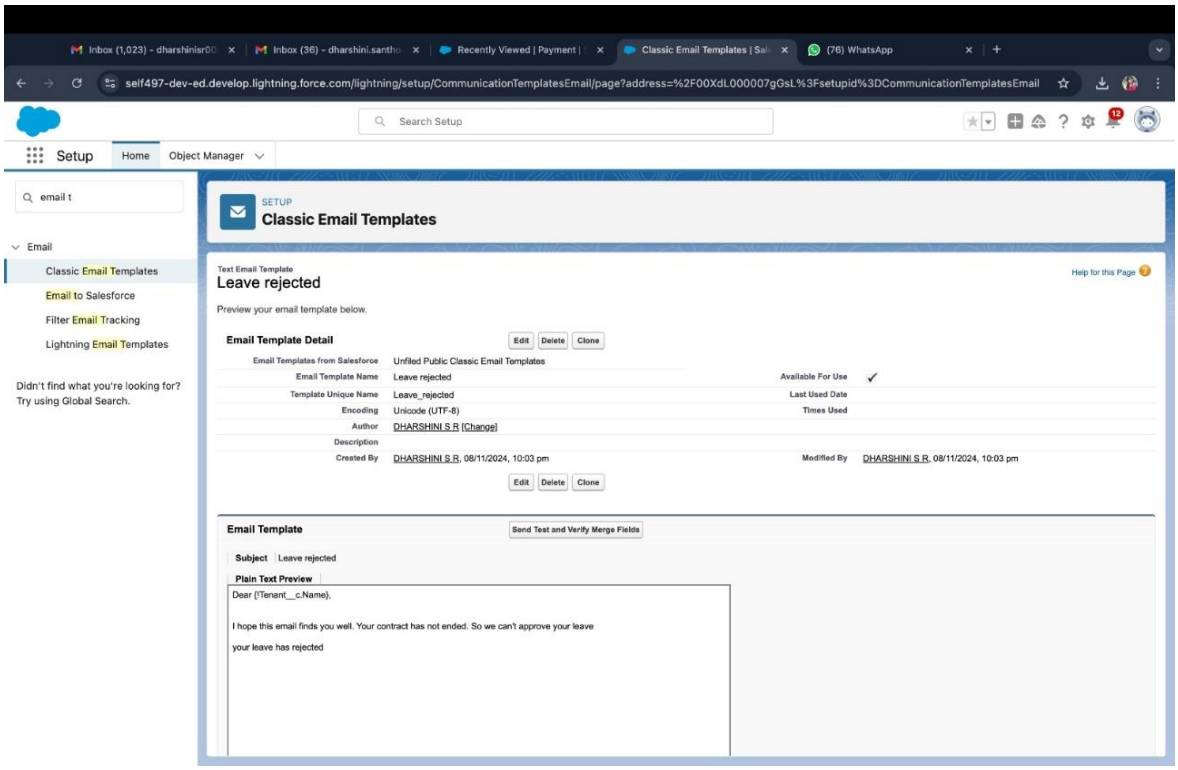
Email Template for Leave Rejection

1. Go to Setup > enter "Email Template" in the Quick Find box > select Classic Email Template.
2. Click New Email Template > Choose Text format.
   - Folder: Unfiled Public Classic Email Templates
   - Check Available for Use.
3. Template Name: Leave Rejected
   - Unique Name: Auto-populated
4. Subject: Leave Rejected
5. Email Body:

   Dear {!Tenant__c.Name},

   Your contract has not yet ended, and we are unable to approve your leave request at this time.

6. Click Save.



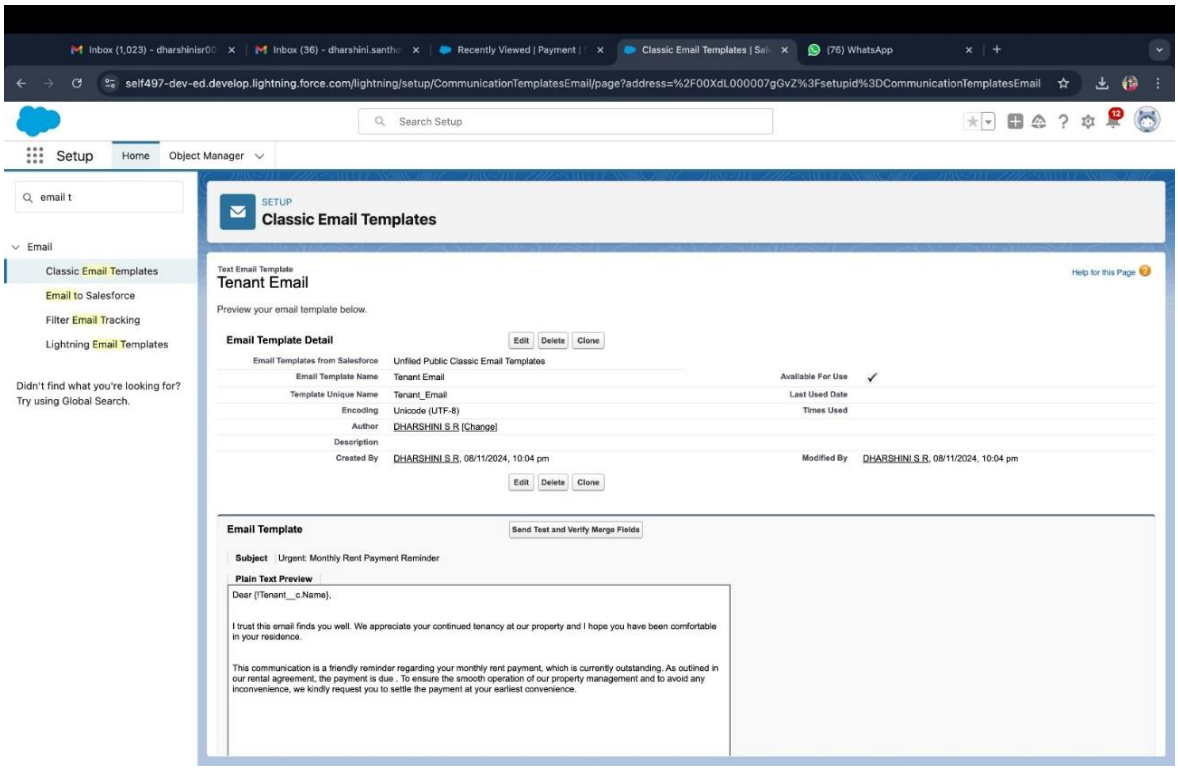
Email Template for Monthly Payment Reminder

1. Go to Setup > enter "Email Template" in the Quick Find box > select Classic Email Template.
2. Click New Email Template > Choose Text format.
   - Folder: Unfiled Public Classic Email Templates
   - Check Available for Use.
3. Template Name: Tenant Email
   - Unique Name: Auto-populated
4. Subject: Urgent: Monthly Rent Payment Reminder
5. Email Body:

   Dear {!Tenant__c.Name},

   This is a friendly reminder about your outstanding monthly rent payment. Please make the payment at your earliest convenience as per our rental agreement.

6. Click Save.



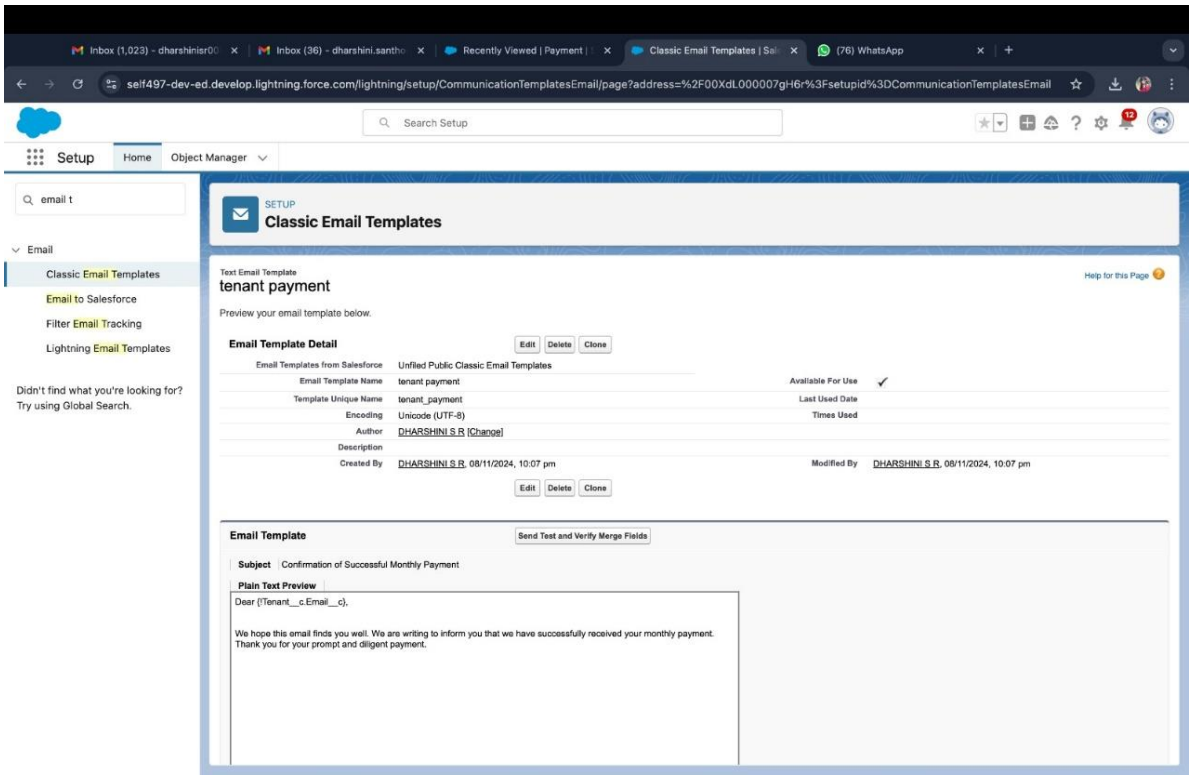
Email Template for Successful Payment Confirmation

1. Go to Setup > enter "Email Template" in the Quick Find box > select Classic Email Template.
2. Click New Email Template > Choose Text format.
   - Folder: Unfiled Public Classic Email Templates
   - Check Available for Use.
3. Template Name: Tenant Payment
   - Unique Name: Auto-populated
4. Subject: Confirmation of Successful Monthly Payment
5. Email Body:

   Dear {!Tenant__c.Email__c},

   We are pleased to inform you that we have received your monthly payment. Thank you for your promptness.

6. Click Save.



8. Approval Process

1. Create Approval Process for "Check for Vacant"

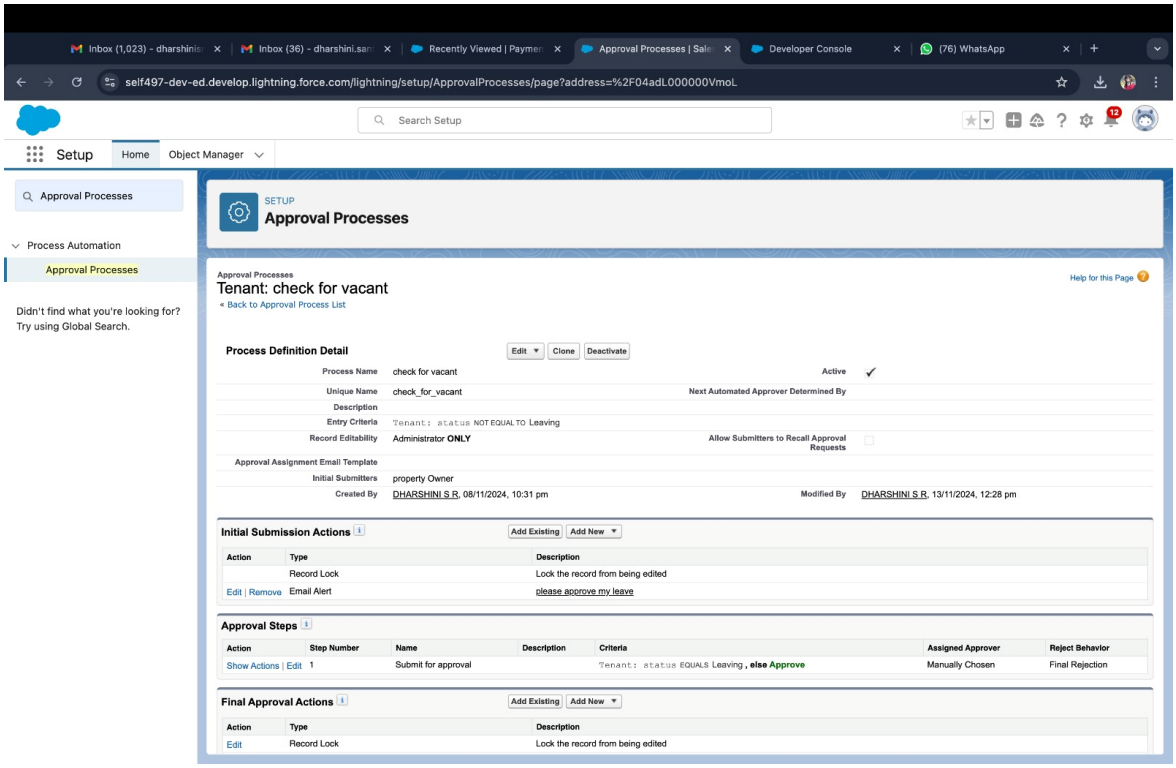
Purpose: This approval process will manage tenant requests by checking the vacancy status, notifying approvers, and taking appropriate actions based on approval or rejection.

Steps:

1. **Navigate to Approval Processes**:
   - Go to **Setup**.
   - In the **Quick Find** box, enter **Approval Processes** and select it.
2. **Choose Object**:
   - Under **Manage Approval Processes For**, select **Tenant** from the dropdown.
3. **Initiate a New Approval Process**:
   - Click **Create New Approval Process** and choose **Use Standard Setup Wizard**.
   - Enter the **Process Name** as **"Check for Vacant"**.
   - Click **Next**.
4. **Set Entry Criteria**:
   - **Field**: Choose **Tenant: Status**.
   - **Operator**: Not Equals.
   - **Value**: Select **Leaving**.
   - Click **Insert Field**, then **Next**.
5. **Define Approval Step**:
   - Set **Next Automated Approver determined by** to **None**.
   - Enable **Administrators ONLY can edit records during the approval process**.
   - Click **Next**.
6. **Skip Approver Email Template**:
   - Leave the email template selection blank and click **Next**.
7. **Configure Fields for Approval Page**:
   - Select **Tenant Name** from **Available Fields** and move it to **Selected Fields**.
   - Click **Next**.
   - Ensure **Display Approver History** is checked.
   - Under **Security Settings**, select **Allow approvers to access the approval page only from within the Salesforce application (Recommended)**.
8. **Specify Allowed Submitters**:

- **Submitter Type**: Select **Owner**.
- **Allowed Submitters**: Select **Property Owner**.
- Click **Next** and **Save**.

9. **Return to Approval Processes Listing**:
    - Select **I'll do this later. Take me back to the listing of all approval processes for this object**.

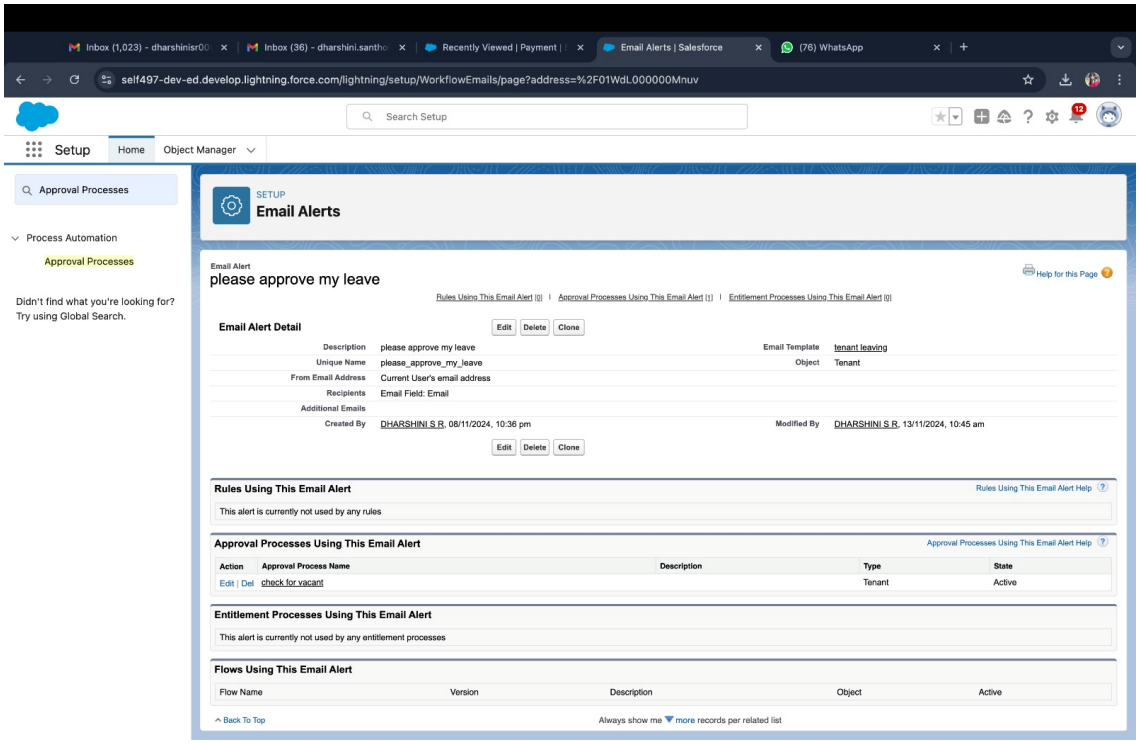


2. Initial Submission Action:

Purpose: Automatically sends an email notification upon initial submission of the leave request.

Steps:

1. Under **Initial Submission Action**, select **Add New** and choose **Email Alert**.
2. Fill out the details:
    - **Description**: "Please approve my leave."
    - **Unique Name**: Auto-populated.
    - **Email Template**: Tenant Leaving.
    - **Recipient Type**: Email Field.
    - **Available Recipients**: Choose **Email Field: Email**.
    - **From Email Address**: Select **Current User's Email**.
3. Click **Save**.

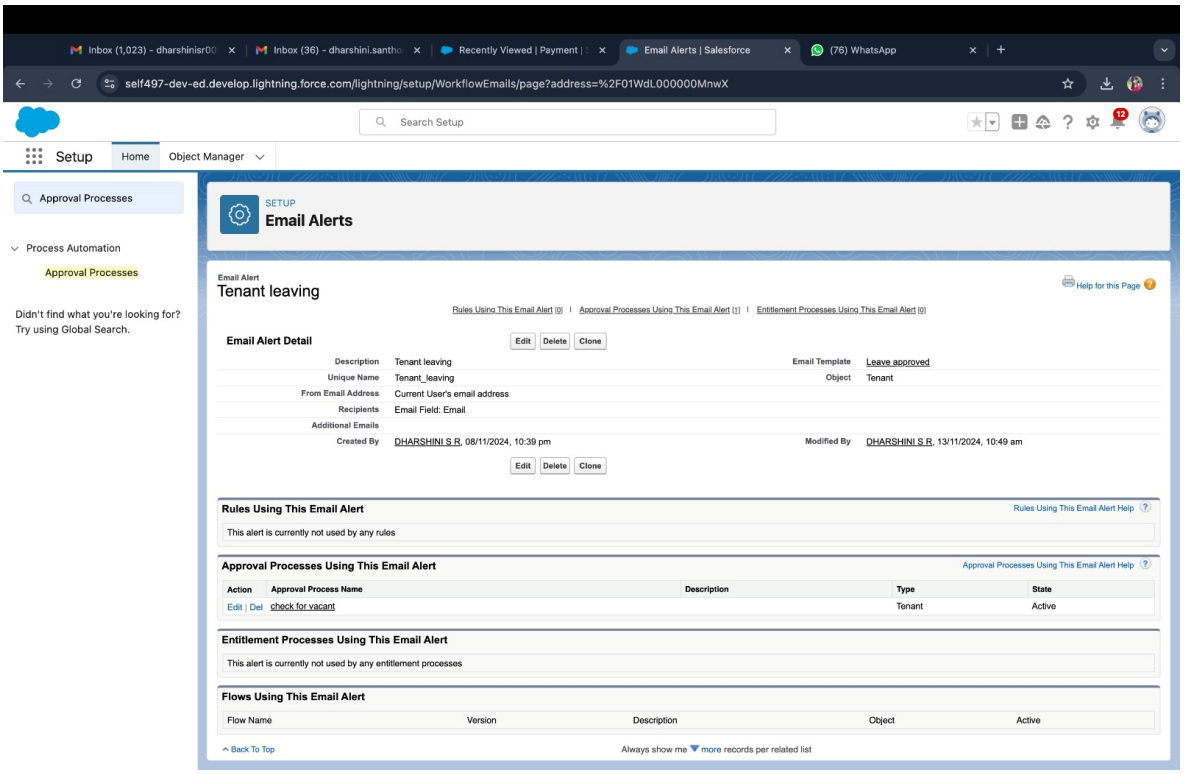


3. Final Approval Action

Purpose: Sends a confirmation email when the tenant's leave request is approved.

Steps:

1. Under **Final Approval Action**, select **Add New** and choose **Email Alert**.
2. Complete the following fields:
    - **Description**: "Tenant Leaving."
    - **Unique Name**: Auto-populated.
    - **Email Template**: Leave Approved.
    - **Recipient Type**: Email Field.
    - **Available Recipients**: Choose **Email Field: Email**.
    - **From Email Address**: Select **Current User's Email**.
3. Click **Save**.

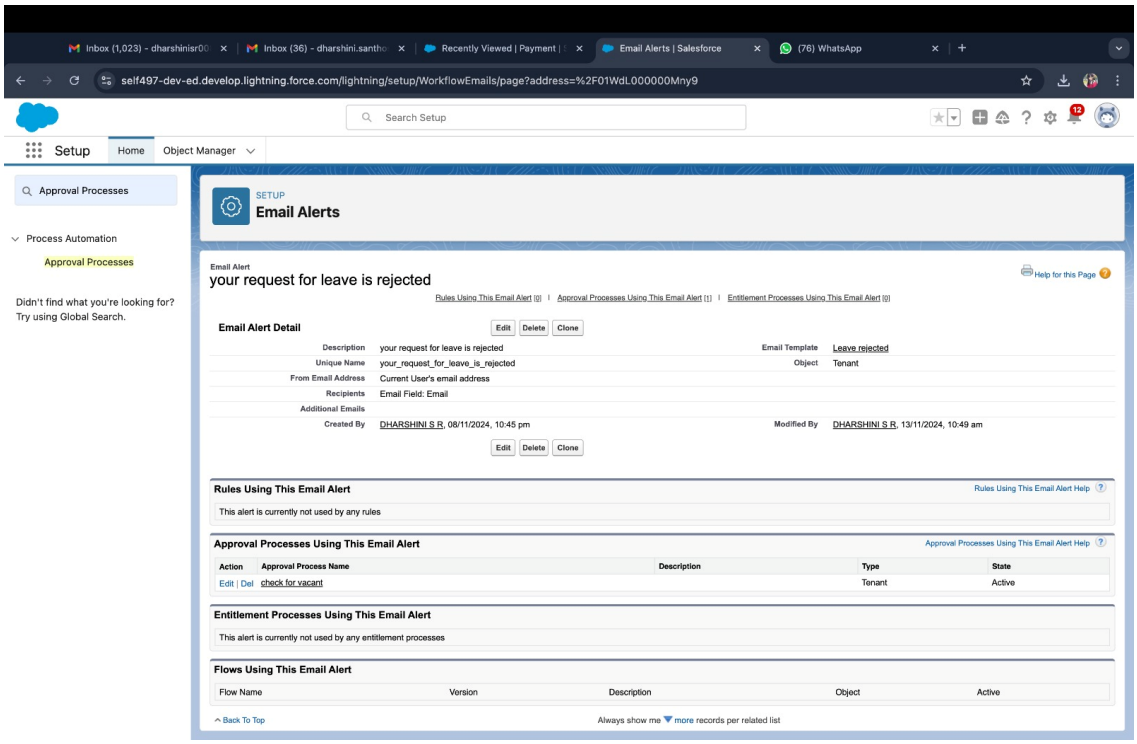


4. Final Rejection Action

Purpose: Notifies the tenant that their leave request has been rejected.

Steps:

1. Under **Final Rejection Action**, select **Add New** and choose **Email Alert**.
2. Fill in the following information:
    - **Description**: "Your request for leave is rejected."
    - **Unique Name**: Auto-populated.
    - **Email Template**: Leave Rejected.
    - **Recipient Type**: Email Field.
    - **Available Recipients**: Choose **Email Field: Email**.
    - **From Email Address**: Select **Current User's Email**.
3. Click **Save**.



9. Apex Trigger

Create the Apex Trigger

i. Go to Setup.

ii. In the Quick Find box, type "Apex Triggers" and select Apex Triggers.

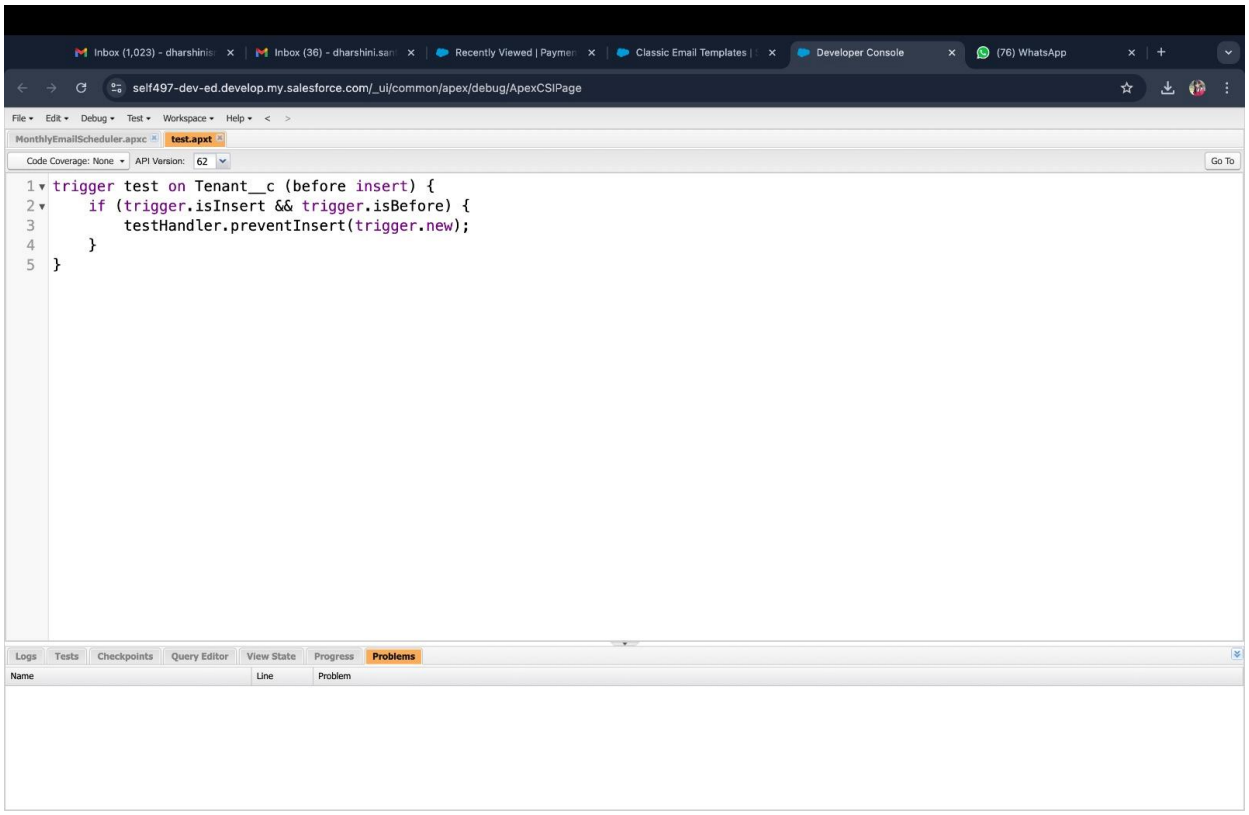
iii. Click New to create a new trigger.

iv. Choose the Tenant__c object from the dropdown and name your trigger test.

v. Enter the following code for the Apex trigger:

```
trigger test on Tenant__c (before insert) {

   // Check if the trigger is for insert and is before saving
   if(trigger.isInsert && trigger.isBefore){
      testHandler.preventInsert(trigger.new);
   }
}
```



Create the Apex Handler Class

i. Go to Setup.

ii. In the Quick Find box, type Apex Classes and click Apex Classes.

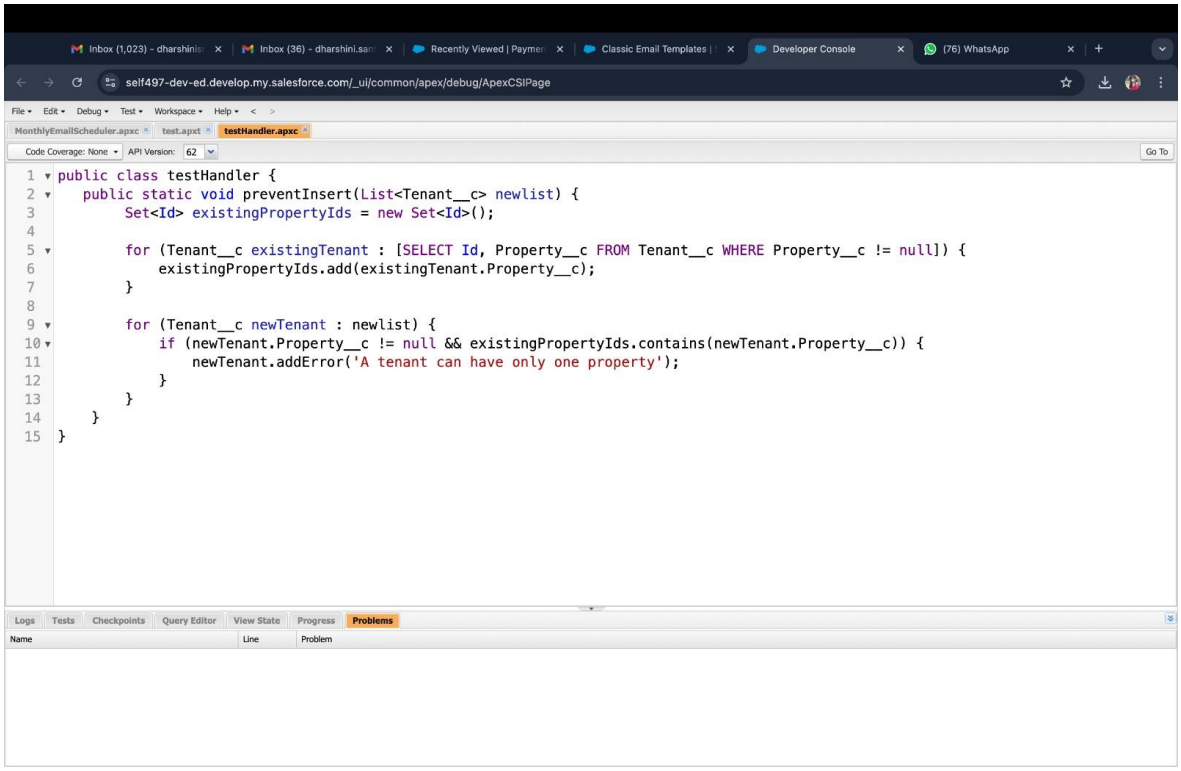
iii. Click New to create a new Apex class and name it testHandler.

iv. Enter the following code logic for the handler class:

```
public class testHandler {
  public static void preventInsert(List<Tenant__c> newlist) {
      Set<Id> existingPropertyIds = new Set<Id>();
      for (Tenant__c existingTenant : [SELECT Id, Property__c FROM
Tenant__c WHERE Property__c != null]) {
          existingPropertyIds.add(existingTenant.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');
          }
      }
  }
}
```



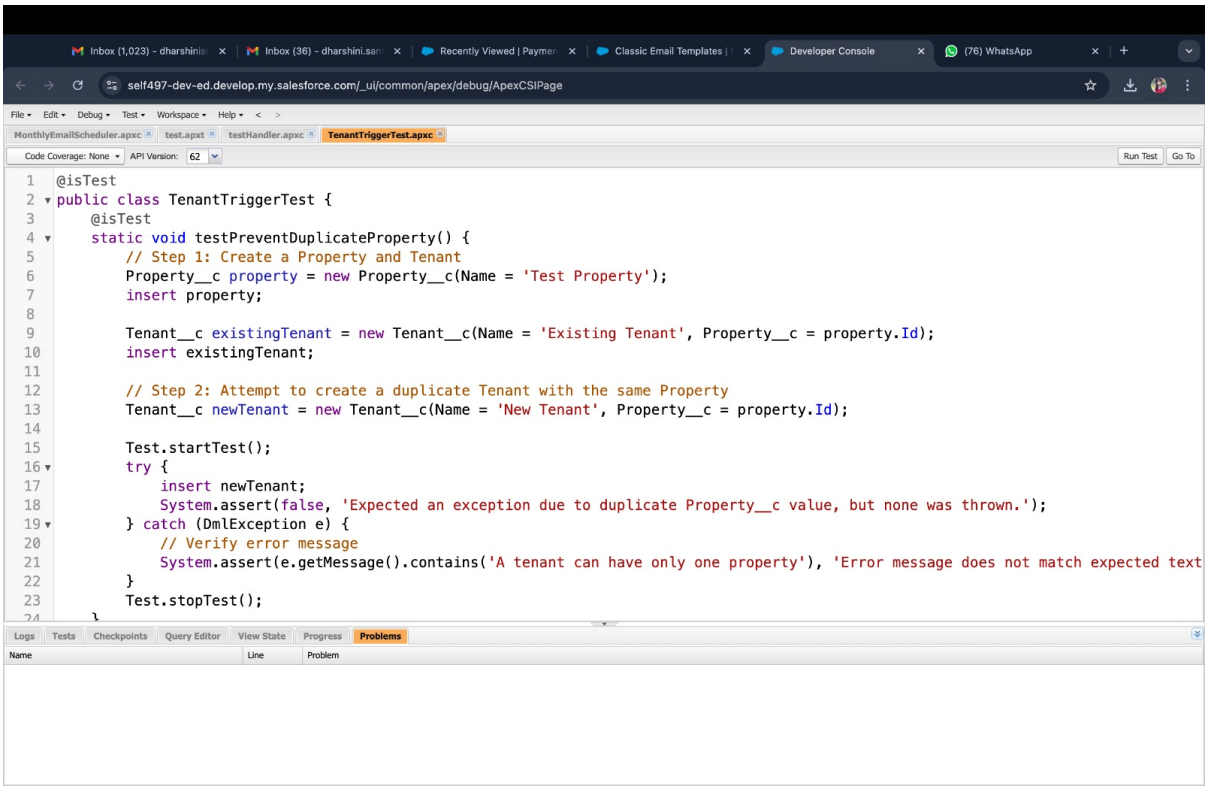

```apex
public class testHandler {
    public static void preventInsert(List<Tenant__c> newlist) {
        Set<Id> existingPropertyIds = new Set<Id>();

        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');
            }
        }
    }
}
```

Testing the Trigger

- ✓ Go to the Tenant tab.
- ✓ Click New to create a new Tenant record.
- ✓ Select a Property that already has an existing tenant.
- ✓ When you try to save, you should see the error message: "A tenant can have only one property".
- ✓ Try creating a Tenant record with a unique Property to confirm it works without errors.



10. Flows

Creating a Monthly Payment Flow

This flow automates the notification process for tenants regarding successful monthly payment confirmations in Salesforce.

Steps:

1. **Navigate to Flow Setup**
   - Go to **Setup**.
   - In the **Quick Find** box, type **Flow** and select **Flows**.
   - Click on **New Flow**.
2. **Choose Flow Type**
   - Select **Record-Triggered Flow**.
   - Click on **Create**.
3. **Set Flow Object and Trigger**
   - Under **Object**, select **Payment for Tenant**.
   - Choose **Trigger**: "A record is updated".
4. **Configure Entry Conditions**
   - Set **Condition Requirements** to **All Conditions Are Met**.
   - Configure the condition as follows:

- **Field**: check_for_payment__c
- **Operator**: Equals
- **Value**: Paid

- Choose **Trigger Frequency**: "Every time a record is updated and meets the condition requirements".

5. **Set Flow Actions and Related Records**
   - Select **Actions and Related Records** and click **Done**.
6. **Add Action to Send Email Notification**
   - In the flow editor, click on the "+" icon and select **Action**.
   - In the **Action Search**, type **Send Email** and select it.
   - Configure the email action as follows:
     - **Label**: Send Email
     - **API Name**: send_email
7. **Define Email Body Resource**
   - Enable **Body** and create a new resource:
     - **Resource Type**: Text Template
     - **API Name**: emailbody
     - **Body**:

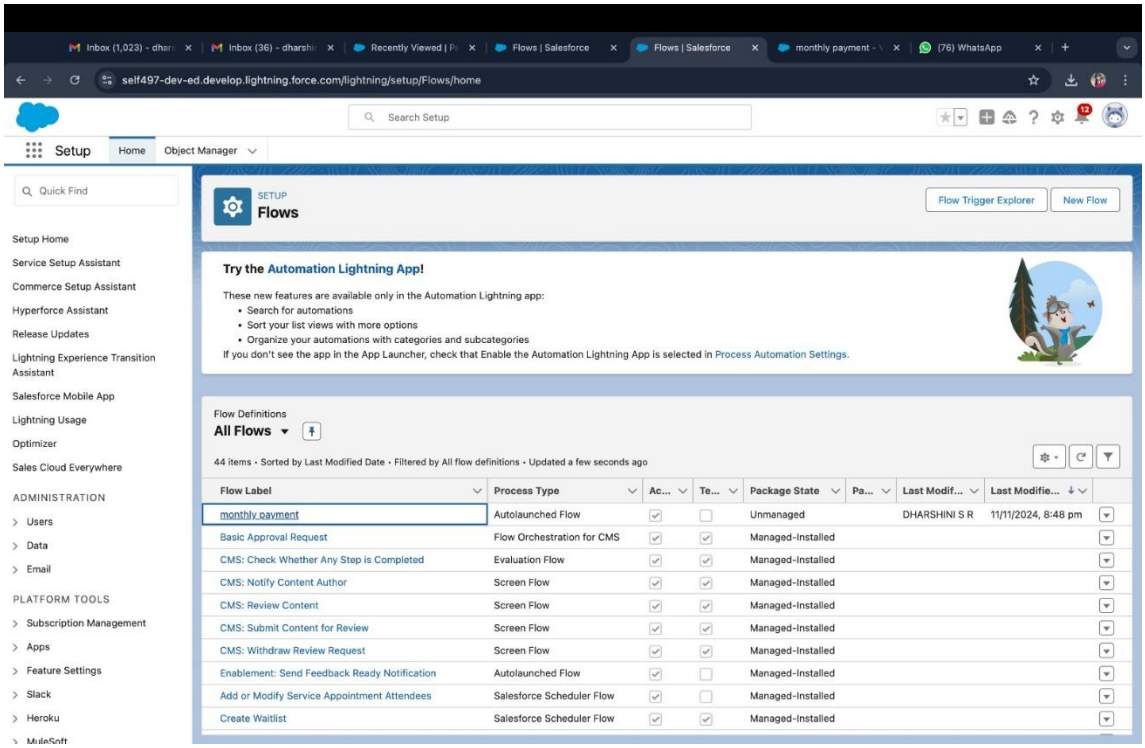
     Dear {!$Record.Tenant__r.Name},

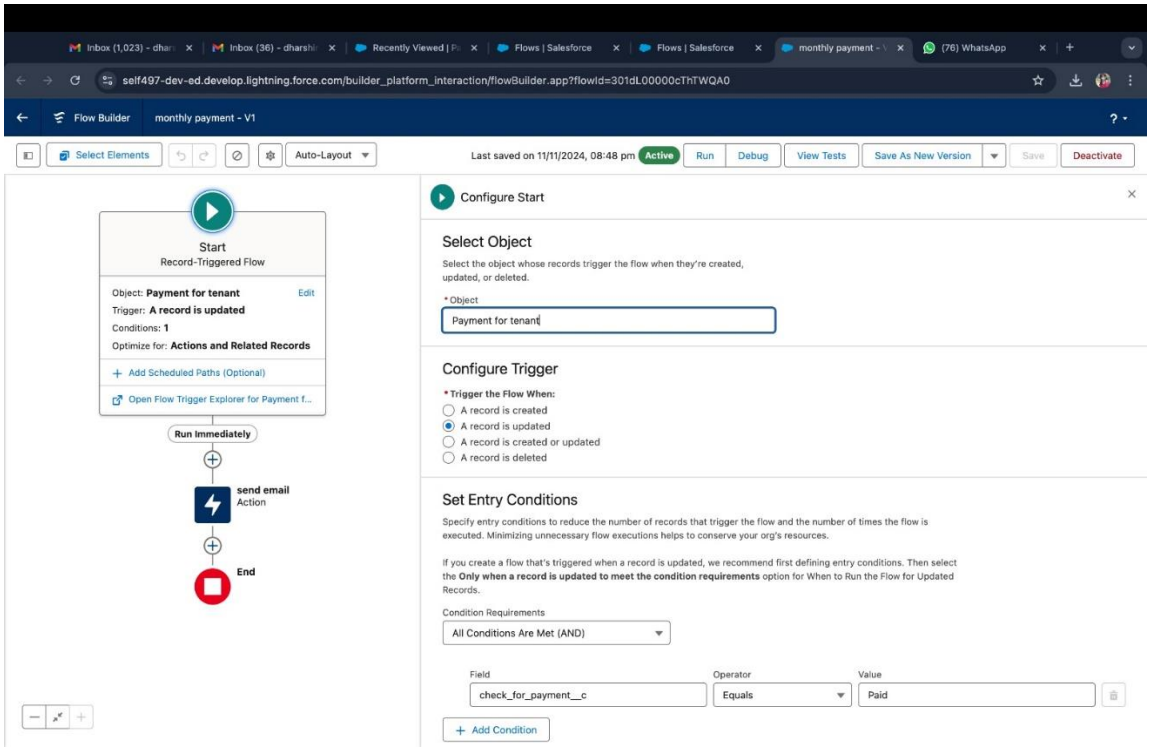
     We hope this email finds you well. We are writing to inform you that we have successfully received your monthly payment. Thank you for your prompt and diligent payment.
   - Click **Done**.
8. **Configure Recipient Address**
   - Enable **Recipient Address List** and paste this:

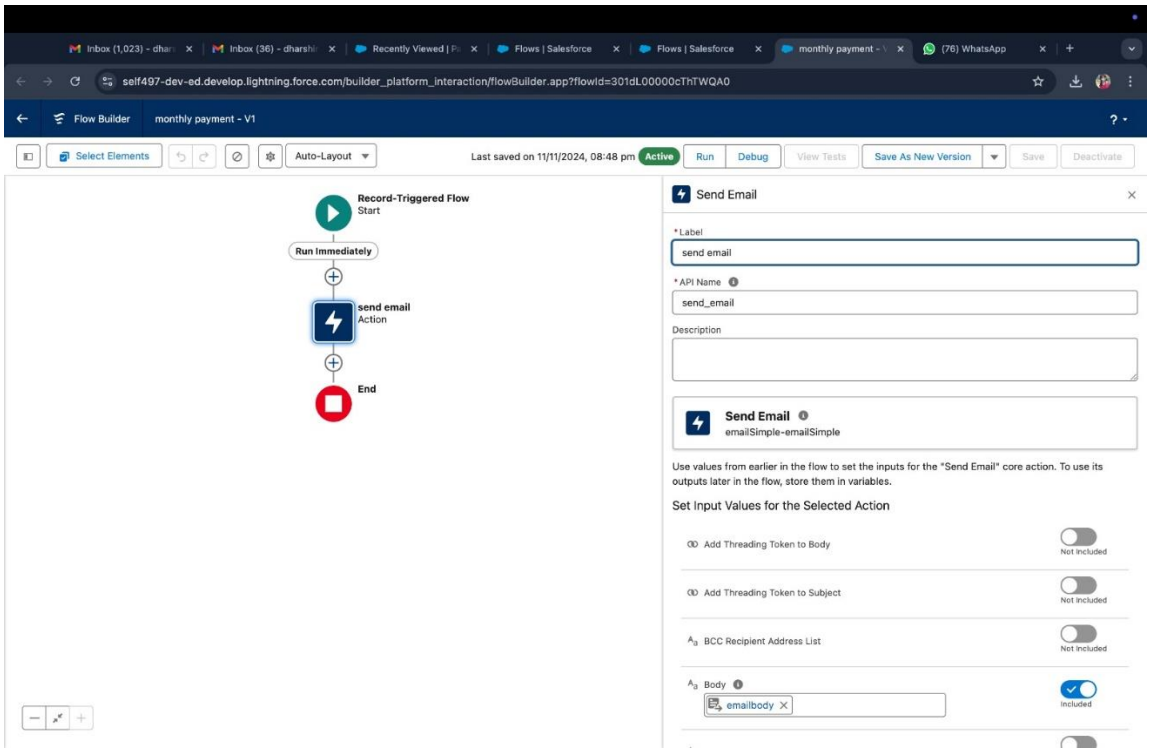
     {!$Record.Tenant__r.Email__c}
   - Click **Done**.
9. **Set Email Subject**
   - Enable **Subject** and paste:

     Confirmation of Successful Monthly Payment
10. **Save and Activate Flow**
    - **Flow Label**: Monthly Payment
    - **Flow API Name**: monthly_payment

- Click **Save**, then **Activate**.







11. Schedule the Apex Class

1. Go to Setup and search for Apex Classes in the Quick Find box.

2. Select Schedule Apex.

3. Fill in the following details:

    Job Name: MonthlyEmailScheduler.

    Apex Class: MonthlyEmailScheduler.

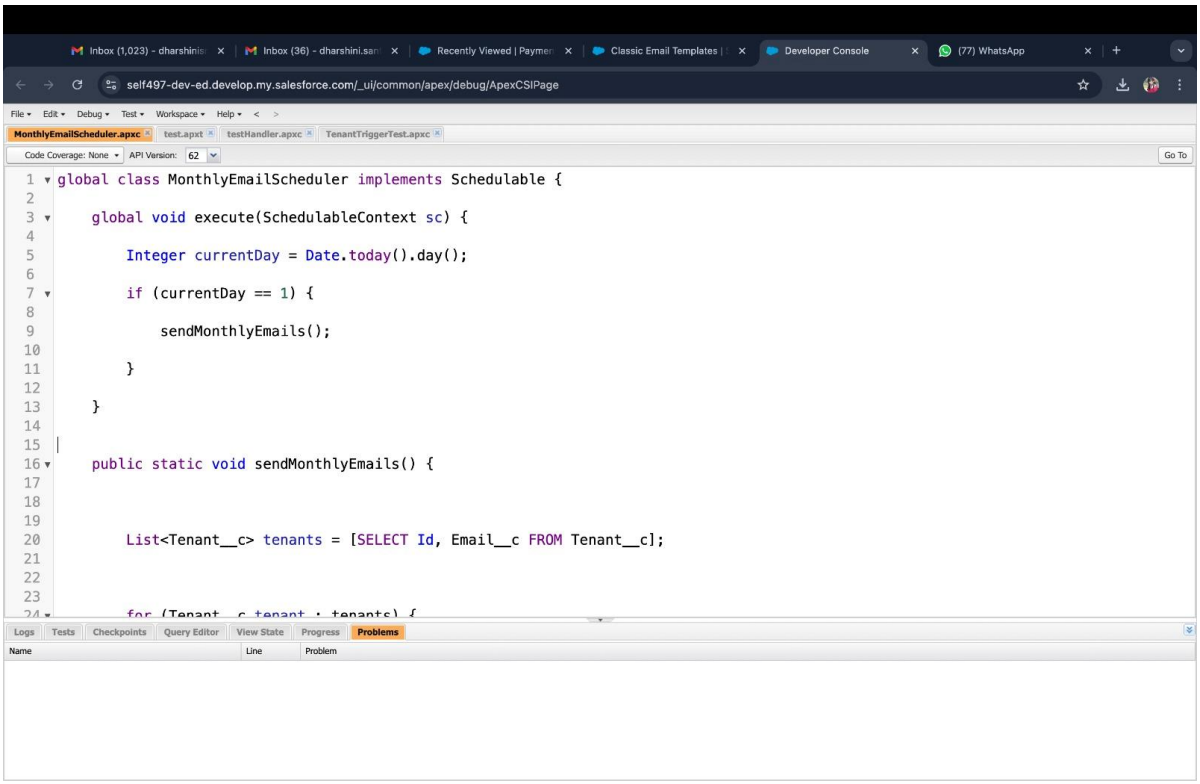
    Frequency: Monthly → Select on Day 1.

    Start Date: Set a start date, e.g., 04/12/2023.

    End Date: Set an end date, e.g., 04/01/2024 (or leave it empty for indefinite).

    Preferred Start Time: Select 09:00 AM.

4. Click Save.



12. Set Up Approval Process (For Lease Agreements or Rent Approvals)

1. Go to Setup and search for Approval Processes in the Quick Find box.

2. Select the Tenant object or relevant object to set up the approval process.

3. Click New Approval Process.

4. Select Use Standard Setup Wizard.

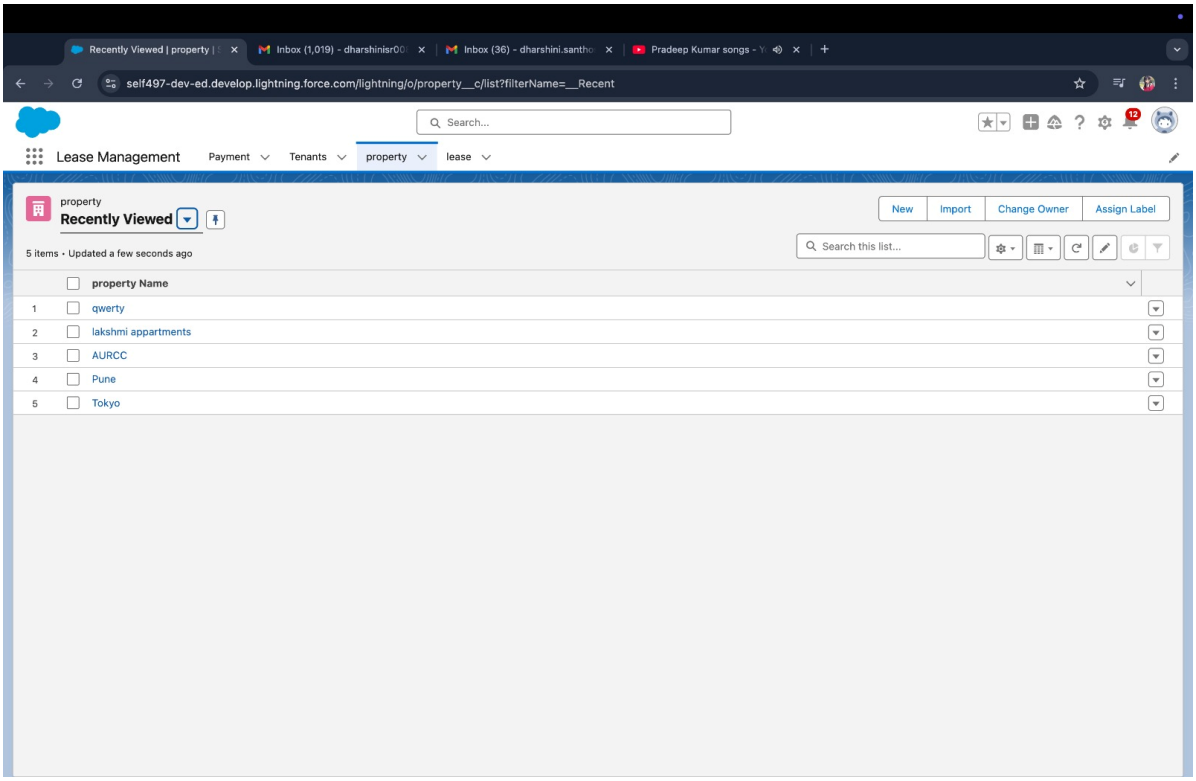
5. Configure the following:

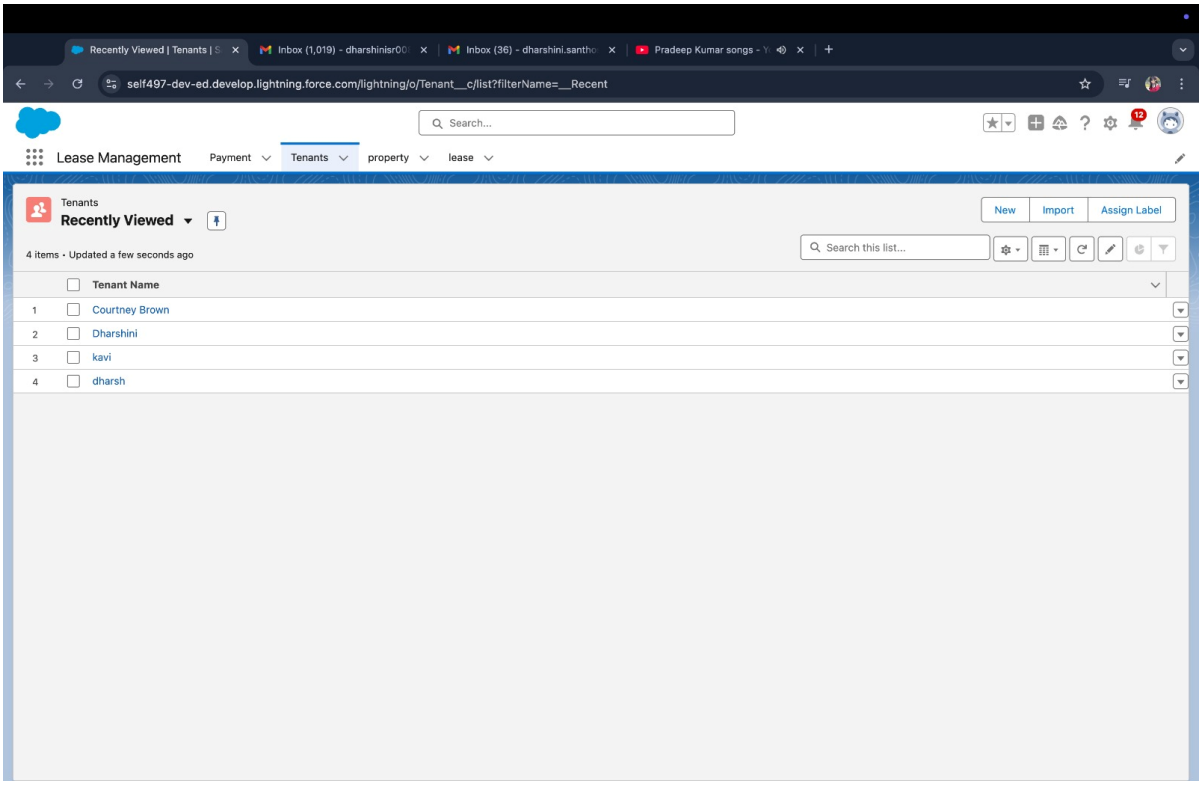
   Entry Criteria: Define the criteria when a record enters the approval process (e.g., when a new lease agreement is created or when a rent payment needs approval).

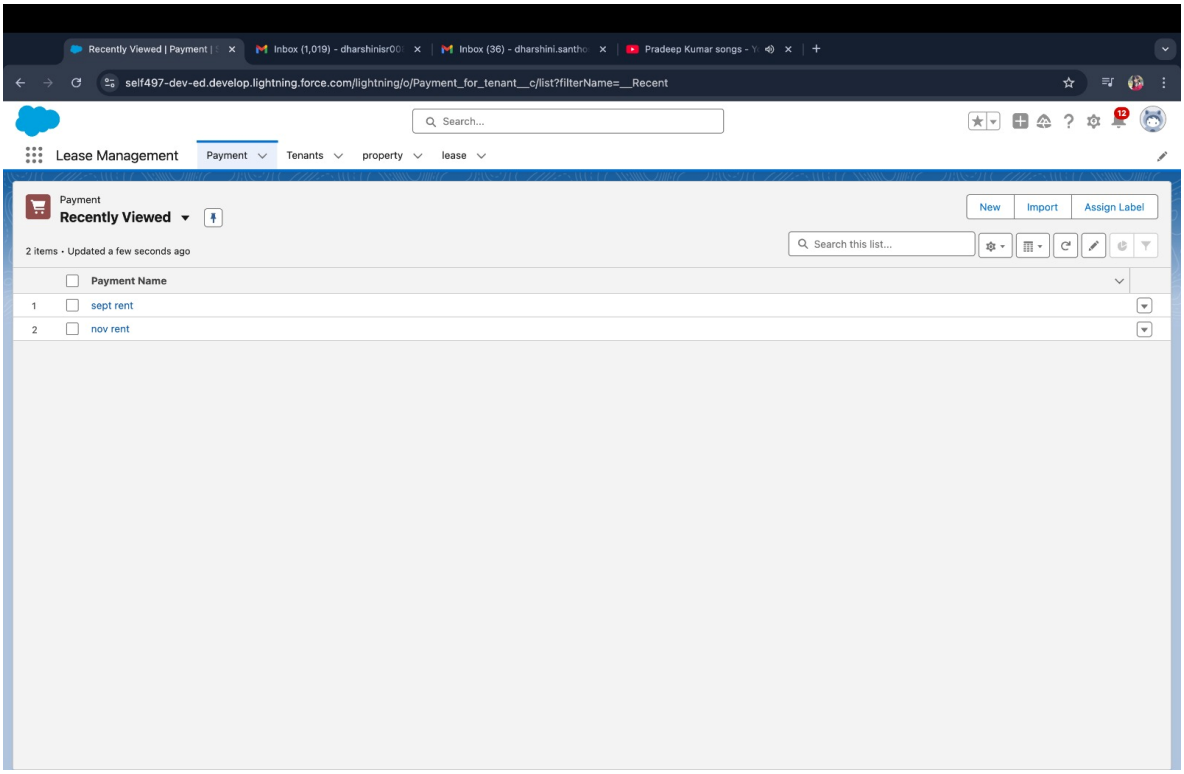
   Approval Steps: Define the steps, approvers, and actions (e.g., an approver reviews the lease and either approves or rejects it).

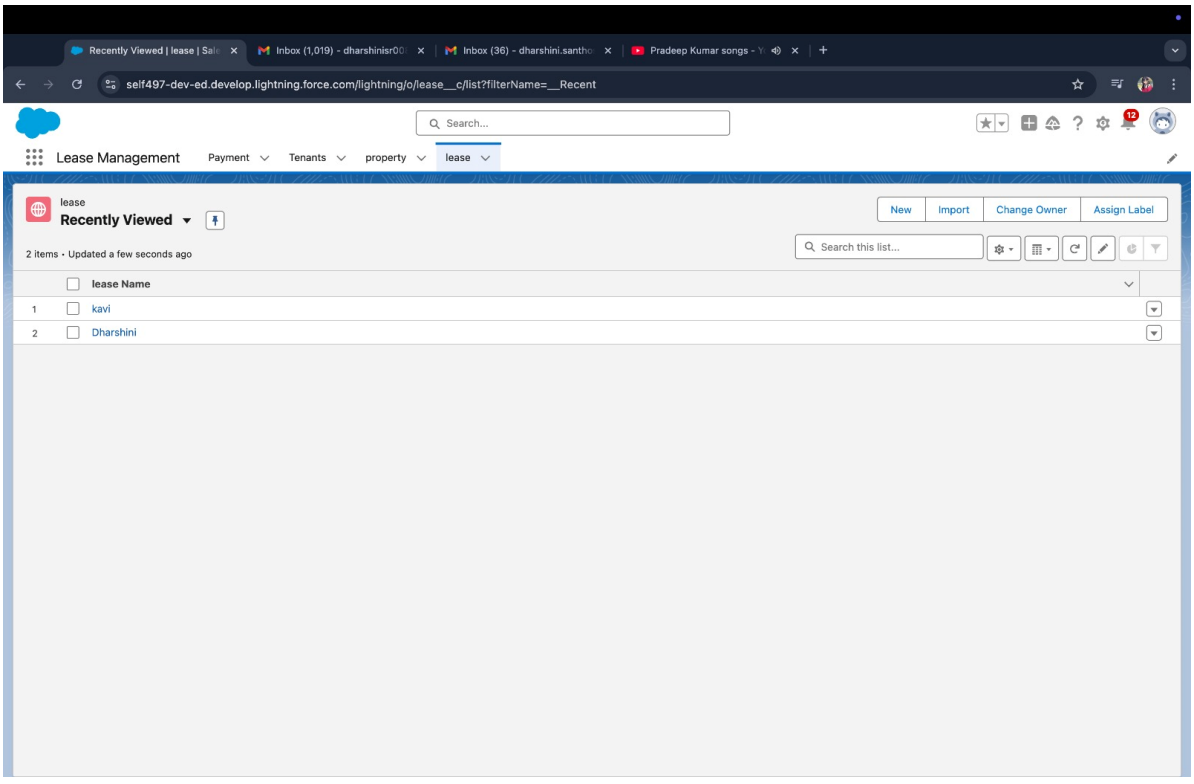
   Final Approval Actions: For example, send a notification email when approved.

6. Save and activate the approval process.


Recently Viewed | property |
Inbox (1,019) - dharshinisr00
Inbox (36) - dharshini.santho
Pradeep Kumar songs -
self497-dev-ed.develop.lightning.force.com/lightning/o/property__c/list?filterName=__Recent
Search...
Lease Management
Payment
Tenants
property
lease
property
Recently Viewed
New
Import
Change Owner
Assign Label
5 items • Updated a few seconds ago
Search this list...
property Name
1 qwerty
2 lakshmi appartments
3 AURCC
4 Pune
5 Tokyo


Recently Viewed | Tenants |
Inbox (1,019) - dharshinisr00
Inbox (36) - dharshini.santho
Pradeep Kumar songs -
self497-dev-ed.develop.lightning.force.com/lightning/o/Tenant__c/list?filterName=__Recent
Search...
Lease Management
Payment
Tenants
property
lease
Tenants
Recently Viewed
New
Import
Assign Label
4 items • Updated a few seconds ago
Search this list...
Tenant Name
1 Courtney Brown
2 Dharshini
3 kavi
4 dharsh


Recently Viewed | Payment
Inbox (1,019) - dharshinisr00
Inbox (36) - dharshini.santho
Pradeep Kumar songs - Y
self497-dev-ed.develop.lightning.force.com/lightning/o/Payment_for_tenant__c/list?filterName=__Recent
Search...
Lease Management
Payment
Tenants
property
lease
Payment
Recently Viewed
New
Import
Assign Label
2 items • Updated a few seconds ago
Search this list...
Payment Name
1
sept rent
2
nov rent


Recently Viewed | lease | Sale
Inbox (1,019) - dharshinisr00
Inbox (36) - dharshini.santho
Pradeep Kumar songs - Y
self497-dev-ed.develop.lightning.force.com/lightning/o/lease__c/list?filterName=__Recent
Search...
Lease Management
Payment
Tenants
property
lease
lease
Recently Viewed
New
Import
Change Owner
Assign Label
2 items • Updated a few seconds ago
Search this list...
lease Name
1
kavi
2
Dharshini

13. Testing the Email Scheduler and Approval Process

Testing the Monthly Email Scheduler:

1. Ensure that the MonthlyEmailScheduler is scheduled to run on the 1st of each month.

2. Go to Developer Console and create some test Tenant records with valid email addresses.

3. Manually trigger the MonthlyEmailScheduler to check that emails are being sent to tenants on the 1st of the month:

You can also use Apex to manually execute the scheduler (System.schedule()).

14. Confirm the emails are being sent (you can check the debug logs or use Test Email tools to confirm).

Testing the Approval Process:

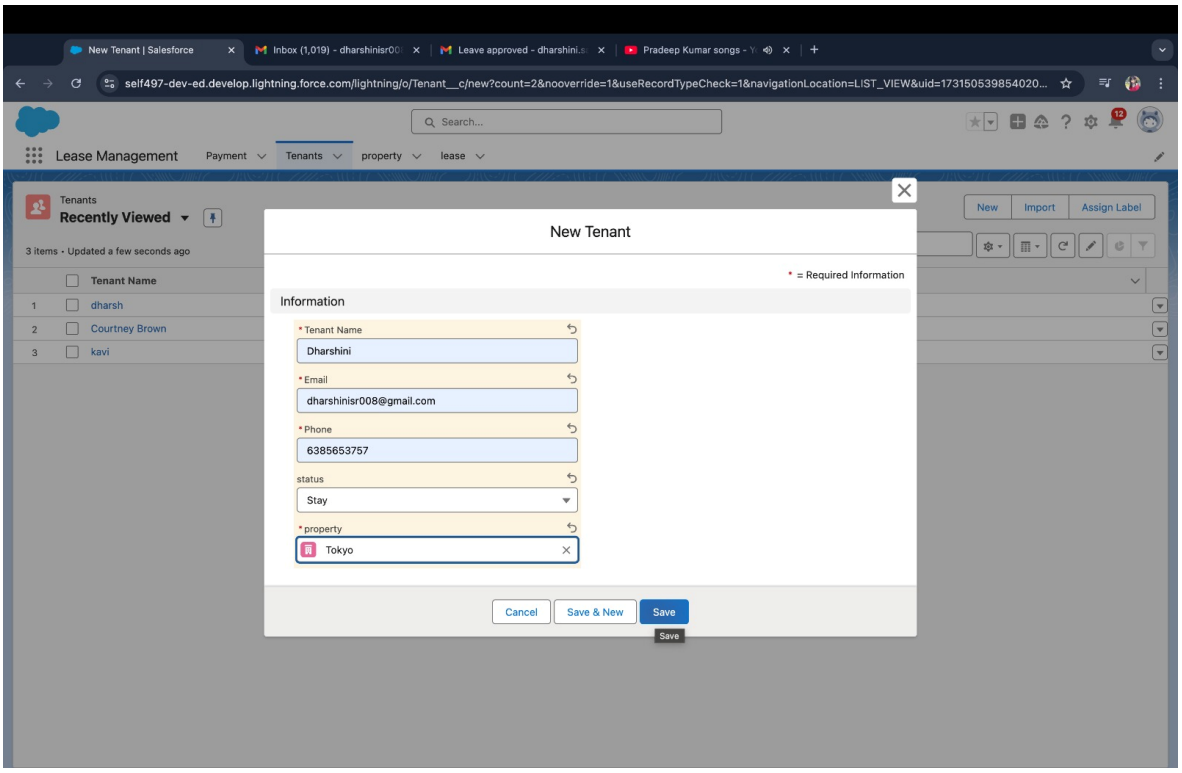
1. Create a new Tenant record or trigger the approval process.

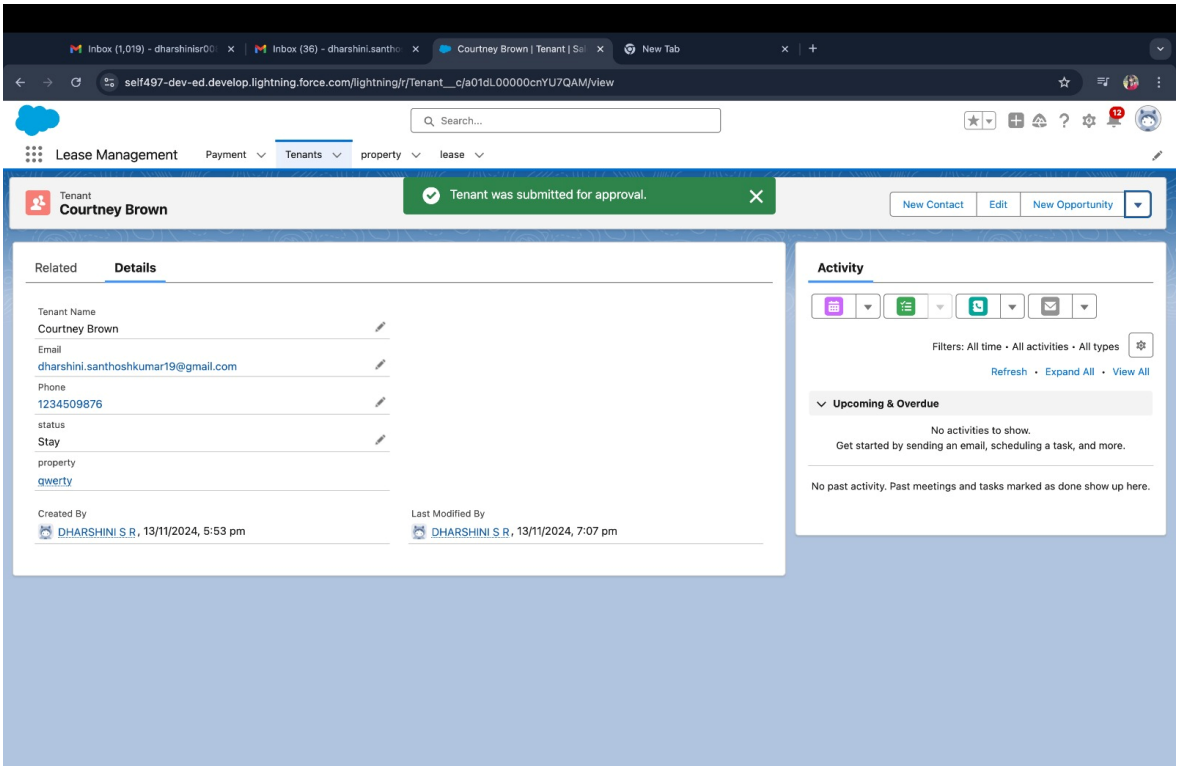
2. Submit the record for approval.

3. As the approver, go to Approval Requests and click on the record you need to approve or reject.

4. Click Approve or Reject and enter any comments.

5. The system will send an email notification based on the approval or rejection status.


New Tenant | Salesforce
Inbox (1,019) - dharshinisr00
Leave approved - dharshini.s
Pradeep Kumar songs -
self497-dev-ed.develop.lightning.force.com/lightning/o/Tenant__c/new?count=2&nooverride=1&useRecordTypeCheck=1&navigationLocation=LIST_VIEW&uid=1731505398540...
Search...
Lease Management
Payment
Tenants
property
lease
Tenants
Recently Viewed
3 items • Updated a few seconds ago
Tenant Name
1 dharsh
2 Courtney Brown
3 kavi
New
Import
Assign Label
New Tenant
* = Required Information
Information
* Tenant Name
Dharshini
* Email
dharshinisr008@gmail.com
* Phone
6385653757
status
Stay
* property
Tokyo
Cancel
Save & New
Save
Save

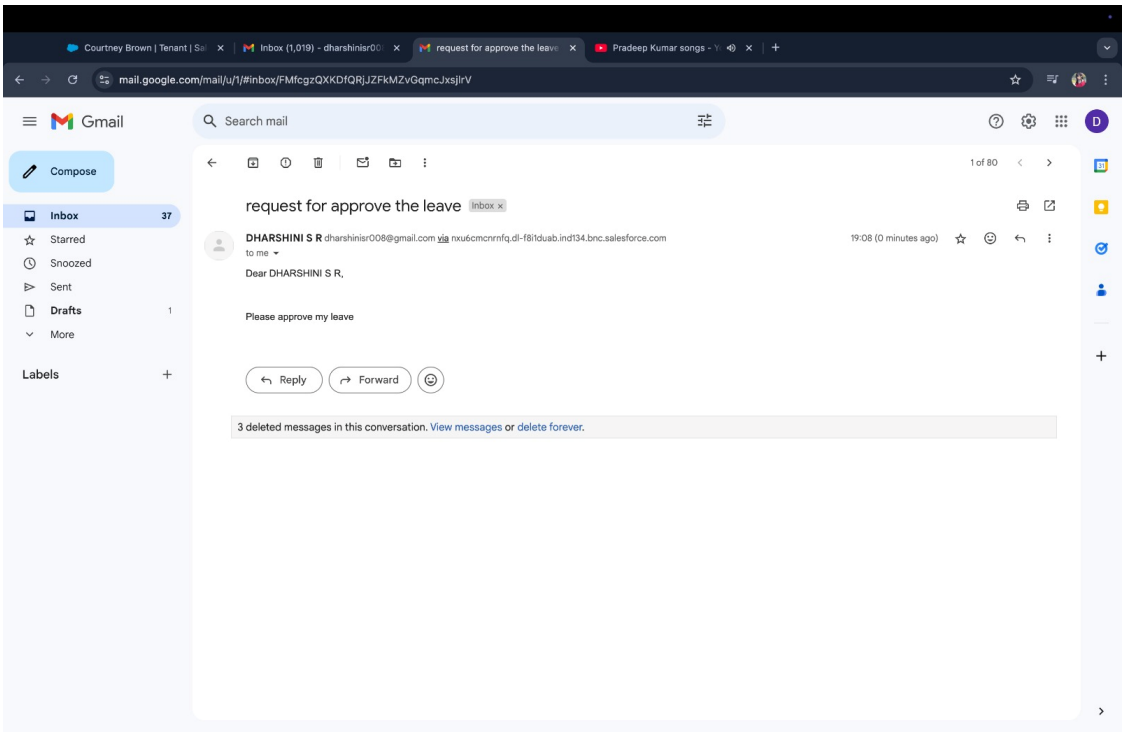

Inbox (1,019) - dharshinisr00
Inbox (36) - dharshini.santho
Courtney Brown | Tenant | Sal
New Tab
self497-dev-ed.develop.lightning.force.com/lightning/r/Tenant__c/a01dL00000cnYU7QAM/view
Search...
Lease Management
Payment
Tenants
property
lease
Tenant
Courtney Brown
Tenant was submitted for approval.
New Contact
Edit
New Opportunity
Related
Details
Tenant Name
Courtney Brown
Email
dharshini.santhoshkumar19@gmail.com
Phone
1234509876
status
Stay
property
qwerty
Created By
DHARSHINI S R, 13/11/2024, 5:53 pm
Last Modified By
DHARSHINI S R, 13/11/2024, 7:07 pm
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.

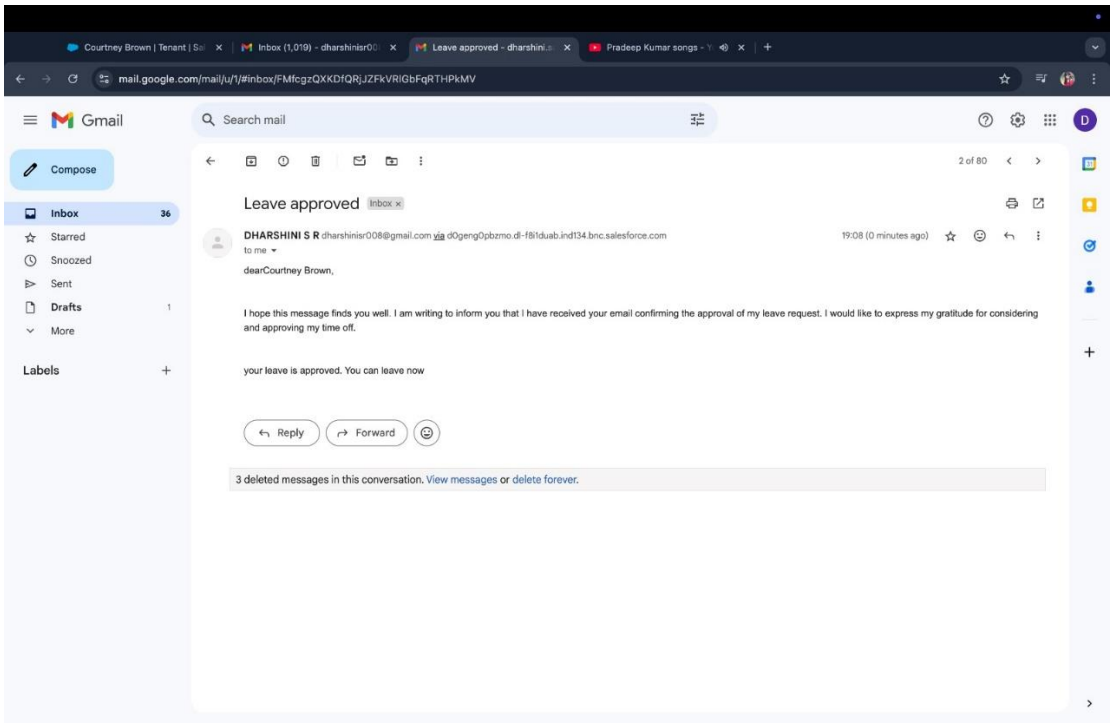
15. Final Testing and Validation

1. Email Testing: Ensure that tenants receive email notifications on the 1st of each month.

2. Approval Process Testing: Submit records for approval and test both approve and reject actions. Verify that notifications and emails are sent correctly.

3. Tenant Record Creation: Create some sample Tenant records to verify the integration with the approval process and email scheduler.





4. Salesforce Key Features and Concepts Utilized

This section outlines the essential Salesforce functionalities and concepts applied in this project, showcasing how each feature contributes to building an efficient and user-friendly system.

1. Custom Objects and Fields

- **Custom Objects**: Created specific custom objects (e.g., Property, Tenant, Payment for Tenant, and Lease) to capture all necessary data unique to the lease management system. Each object represents a key entity within the project, allowing structured data storage and efficient data retrieval.
- **Custom Fields**: Defined fields within each object to capture specific data points, such as Tenant Name, Payment Status, and Lease Details. These fields support data organization and enhance the system's capability to track various attributes relevant to each entity.

2. Approval Processes

- **Automated Workflow for Approvals**: Developed approval processes, such as the "Check for Vacant" process, to standardize the review and approval of key actions, like tenant departure requests. This feature facilitates efficient decision-making and ensures approvals are routed to appropriate stakeholders.
- **Conditional Approval Steps**: Configured criteria-based approval steps (e.g., tenant status check) to determine when a record qualifies for approval. This ensures that only records meeting specific conditions move forward in the process.

3. Email Templates and Alerts

- **Consistent Communication with Templates**: Utilized email templates to streamline communications for various actions (e.g., tenant leaving, leave approval, payment confirmation). Templates save time and ensure consistent messaging by including relevant record data through merge fields.
- **Email Alerts**: Created automated email alerts tied to the approval process and other workflows to inform tenants and stakeholders of status updates. Email alerts enhance communication efficiency by automating message delivery based on specific triggers.

4. Flow Automation

- **Record-Triggered Flows**: Designed flows, such as the monthly payment flow, to automate complex tasks based on record updates. For example, when a payment status is marked as “paid,” the flow automatically sends a confirmation email to the tenant.
- **Actions and Conditions**: Set up flow actions (e.g., send email) with specific entry conditions to execute only when records meet defined criteria. This ensures that only relevant records trigger automation, reducing unnecessary system activity.

5. Record Security and Access Controls

- **Field-Level Security and Profiles**: Configured field-level security to control visibility and edit permissions on sensitive data fields, ensuring that only authorized users can access or modify certain information.
- **Approval Page Security Settings**: Enabled restrictions to allow approvers to access the approval page only from within Salesforce, reinforcing the system's security and controlling where and how approvals are managed.

6. Reporting and Analytics

- **Custom Report Types**: Enabled reporting features on relevant objects to generate insights into tenant activity, payment status, and approval history. Reports provide an overview of key metrics and facilitate decision-making.
- **Dashboard Integration**: Utilized dashboards to provide a visual summary of report data, offering a quick and accessible view of project performance metrics for stakeholders.

7. User Notifications

- **Notification Setup for Approvals**: Configured in-app and email notifications for approval actions to alert users of requests requiring their attention. Notifications are crucial for timely approvals and ensure users are promptly informed of any status changes.

5. Detailed Steps to Solution Design

This section provides a comprehensive outline of the design process, covering the creation of data models, user interface designs, and business logic. Each element is thoroughly documented to provide a clear view of the system architecture and functionality, supplemented by relevant screenshots to illustrate the solution visually.

Step 1: Define Data Models

- **Identify Key Objects and Relationships**: Define custom objects (e.g., Property, Tenant, Payment, Lease) that represent the main entities within the system.
- **Design Data Fields**: Create fields for each object to capture essential data points, ensuring field names, data types, and relationships are consistent with business requirements.
- **Establish Relationships**: Define lookups or master-detail relationships between objects where needed, ensuring data linkage supports the flow of information.

Step 2: Design the User Interface

- **Create Custom Tabs**: Set up tabs for each main object (e.g., Property, Tenant, Lease) to provide easy access within the Salesforce app.
- **Page Layouts**: Configure layouts for each object to display fields and related records

logically, ensuring a user-friendly interface.

- **Approval Process Buttons**: Add buttons on the record pages for submitting approval requests and tracking status to streamline user actions.

Step 3: Define Business Logic

- **Approval Processes**: Develop approval processes for requests such as tenant departure approvals. Define entry criteria, approval steps, and rejection actions.
- **Flows and Automation**: Create flows for automated tasks, such as sending payment confirmation emails. Configure triggers, conditions, and actions to ensure flows execute only when necessary.
- **Validation Rules and Workflows**: Set up validation rules to enforce data accuracy (e.g., ensuring payment status is updated correctly). Define workflows for tasks requiring email alerts or field updates.

Step 4: Documentation of Screens and Processes

- **Screenshots of Each Process Step**: For approval processes, provide screenshots of each step, from initiation to final approval/rejection notifications, to offer a visual guide.
- **Flow Diagrams**: Include diagrams of complex flows to illustrate how data moves through the system. Ensure that each step in the flow is labeled and connected to show dependencies.

6. Testing and Validation

This section outlines the testing strategy used to ensure the Salesforce solution functions as intended and meets business requirements. Our approach to testing incorporates both backend and frontend validations to verify that each component operates accurately and delivers a seamless user experience.

Approach to Testing

1. Unit Testing

- **Objective**: Validate individual pieces of code, such as Apex classes and triggers, to ensure they perform as expected without causing any errors.
- **Scope**:
    - **Apex Classes**: Write and execute unit tests for each Apex class used in the project, covering a wide range of test cases to check class methods, logic, and interactions with custom objects.
    - **Triggers**: Implement unit tests for each trigger to confirm they execute under the defined conditions, manage related records accurately, and handle exceptions appropriately.

- **Code Coverage Requirements**: Ensure that each Apex class and trigger achieves a minimum of 75% code coverage, as required by Salesforce, to validate functionality and support deployment to production.
- **Example Unit Test**: For the "Tenant Departure" approval process, unit tests verify the successful submission and handling of approval requests under various conditions, such as valid vs. invalid tenant status.

2. User Interface Testing

- **Objective**: Assess the user experience, ensuring the solution's front end is intuitive, responsive, and consistent with user requirements.
- **Scope**:
    - **Page Layouts and Tabs**: Test each custom tab (Property, Tenant, Lease, Payment) and page layout for correct display and functionality, ensuring users can view and interact with records effortlessly.
    - **Approval Process Buttons and Notifications**: Confirm the visibility and functionality of approval buttons, the ability to submit records for approval, and receipt of notification alerts for both approved and rejected requests.
    - **Flow and Email Automation Testing**: For flows related to monthly payments, simulate payment submissions to verify the triggering of email templates, checking email content, and recipient accuracy.
    - **Responsiveness**: Ensure the UI components adapt correctly across different devices and screen sizes, providing a consistent user experience on both desktop and mobile.

Test Documentation and Validation

- **Test Scenarios and Expected Results**: Document each test scenario with detailed expected outcomes, including success criteria and handling of edge cases.
- **Test Logs**: Maintain logs of all test executions, including the test results, defects encountered, and any corrective actions taken.
- **Stakeholder Review**: Conduct a user acceptance testing (UAT) phase, where end-users validate that the system aligns with business goals, capturing feedback to address any remaining usability or functionality concerns.

7. Key Scenarios Addressed by Salesforce in theImplementation Project

In this Salesforce implementation project, several key scenarios have been effectively

addressed, showcasing the platform's capabilities in handling complex business processes and automating essential tasks. These scenarios illustrate how Salesforce enhances operational efficiency, improves data visibility, and ensures timely actions in various lease management tasks.

Key Scenarios Addressed

1. Tenant Approval Process for Lease Vacancies

- **Scenario**: When a tenant requests to leave a property, there's a need for an approval process to ensure that the request is reviewed, validated, and approved by relevant property managers.
- **Salesforce Solution**: Using Salesforce's Approval Processes, we created a streamlined process to handle tenant departure requests. This approval process includes email notifications, the option for managers to approve or reject requests, and automated status updates, ensuring consistency and reducing manual errors.

2. Automated Monthly Payment Reminders and Confirmation

- **Scenario**: Regular communication with tenants is essential to remind them of their monthly rent payment and confirm receipt once payment is processed.
- **Salesforce Solution**: Using Flows and Email Templates, the project automates payment reminders for tenants, reducing the need for manual follow-ups. Once payment is received, Salesforce automatically triggers a confirmation email, providing tenants with real-time acknowledgment and maintaining good client relations.

3. Tracking Tenant Status and Lease Details

- **Scenario**: For property managers, keeping track of tenant statuses (active, pending departure, or vacancy) and lease details is crucial for occupancy planning and maintaining accurate records.
- **Salesforce Solution**: Custom objects for Tenant, Lease, and Property, combined with custom fields and page layouts, provide a structured and accessible view of all essential data. This structure allows users to quickly see tenant statuses, lease durations, and property availability, supporting informed decision-making.

4. Dynamic Reporting and Analytics for Financial and Operational Insights

- **Scenario**: Management requires insights into revenue from tenant payments, pending approvals, and vacancy rates to drive data-informed decisions and financial forecasting.
- **Salesforce Solution**: Custom reports and dashboards track payment statuses, approval rates, and vacancy rates, offering real-time analytics. These tools provide decision-makers with actionable insights into financial health and operational metrics, enabling strategic

planning based on accurate data.

5. Streamlined Tenant Communication for Leave Approvals and Rejections

- **Scenario**: When tenants submit a leave request, there needs to be a quick, consistent response to approve or reject the request, with clear communication back to the tenant.
- **Salesforce Solution**: Email Templates for approvals and rejections are automatically triggered by the approval process. This ensures tenants are kept informed of their request status, reducing confusion and enhancing customer satisfaction by providing timely responses.

6. Centralized Management of Property and Lease Records

- **Scenario**: Managing a large portfolio of properties and tenant leases requires a centralized database to handle details efficiently, from lease terms to property characteristics.
- **Salesforce Solution**: Custom objects like Property and Lease create a central repository for storing and retrieving lease agreements, property attributes, and tenant records. This enables property managers to access, update, and manage records from a single source of truth, enhancing data consistency and operational efficiency.

8. Conclusion

Summary of Achievements

The Salesforce implementation project for lease management successfully addressed the primary goals of streamlining property and tenant management, enhancing communication, and automating essential workflows. By leveraging key Salesforce features like Approval Processes, Flows, Email Templates, and Custom Objects, the project provided a comprehensive solution that optimizes daily operations for property managers and improves the tenant experience.

Key Achievements include:

1. **Automated Approval and Notification Processes**: Approval workflows for tenant requests—such as lease terminations and vacancy checks—were automated, saving time and reducing errors. Notifications keep tenants and managers updated in real-time, increasing process transparency.
2. **Efficient Payment Management**: Monthly payment reminders and confirmation emails were fully automated, ensuring timely reminders for tenants and prompt confirmation of successful payments. This automation reduced the need for manual follow-ups and maintained accurate payment records.

3. **Centralized Data and Enhanced Accessibility**: Custom objects for Property, Tenant, Lease, and Payment created a centralized, easily accessible database, improving data visibility and simplifying record management. This centralized data model allowed for improved tracking of tenant statuses, lease terms, and property occupancy.

4. **Improved Decision-Making with Real-Time Reporting**: Custom reports and dashboards provided management with valuable insights into payment statuses, vacancy rates, and tenant retention. These analytics enabled data-driven decisions, better financial planning, and strategic property management.

5. **Consistent Tenant Communication**: With standardized email templates, tenant communication became more efficient and consistent. Tenants received timely and clear responses for all essential interactions, fostering trust and enhancing their experience.

This project demonstrated Salesforce's robust capabilities to handle the complexities of lease management, with achievements that directly impact operational efficiency, tenant satisfaction, and data accuracy. Moving forward, the solution can be further enhanced to integrate more advanced features and adapt to future business needs, ensuring long-term value and scalability.