

# Lease Management

College: 7155 - PSG Institute of Technology and Applied Research

Team ID: NM2024TMID00673

Team Lead: PRAVEEN KUMAR R (18CC27211358BB98C5E4AF89E7B7A93A)

Team Members:

PRAVEEN KUMAR R	18CC27211358BB98C5E4AF89E7B7A93A
KARTHIKEYAN A S	677D4069469C9B7A40E47E97A580ABF5
SRINIVASA PRADEEP S	D95FD874DA7EF00920A4844D3807BEE8
VIJAY RAJ KUMAR M	941B4EC581449338B3A466019EFEC792
RAJASURYA E	77E8E2B9493E4528F0943402CAE00AF6

## 1. Project Overview

This Lease Management project is designed to optimize and automate the processes involved in managing lease agreements for real estate properties, equipment, or other assets. By leveraging Salesforce's platform, the project aims to streamline lease management tasks, improve communication, and ensure compliance with regulations. The system will facilitate efficient tracking, communication, and record-keeping for all lease-related transactions, enhancing both operational efficiency and data accuracy.

## 2. Objectives

### Business Goals:

- To create a streamlined and automated system for lease management that supports efficient tracking, compliance, and tenant relations.

- The project aims to reduce administrative overhead, improve communication with tenants, and ensure compliance with lease terms and regulations.

### **Specific Outcomes:**

- A comprehensive dashboard to track lease terms, payments, and tenant details.
- Automated reminders and notifications for lease renewals, payment due dates, and tenant communications.
- Efficient approval processes and validation checks to ensure data accuracy and compliance.

## **3. Salesforce Key Features and Concepts Utilized**

This project uses key Salesforce functionalities to develop a robust lease management system:

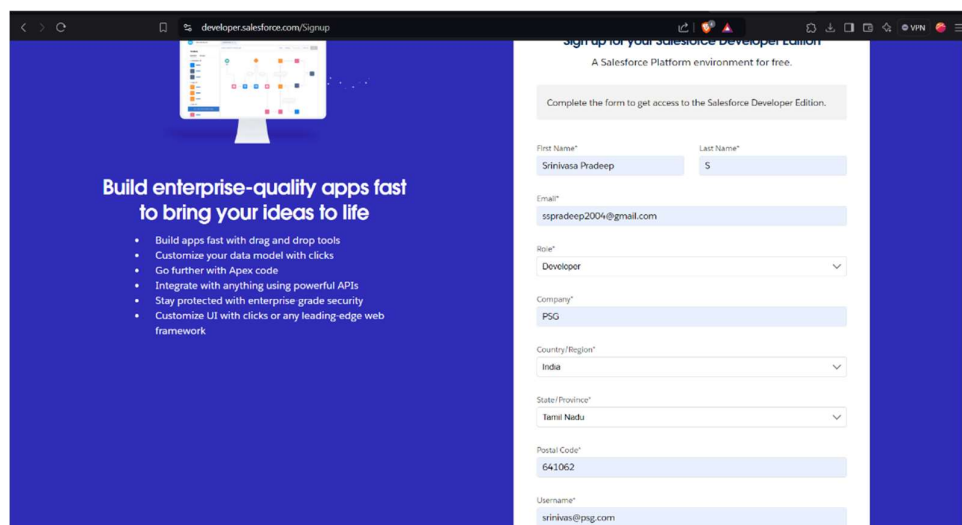
1. Custom Objects and Relationships
  - Created custom objects such as Property, Tenant, Lease, and Payment to manage essential lease data.
  - Established lookup relationships among objects to connect lease records with tenants and properties, supporting easy data retrieval and record linkage.
2. Tabs and Lightning App (Lease Management)
  - Created tabs for each custom object, making navigation easier within the Lease Management Lightning app.
  - Customized the app to organize functionalities for streamlined access and branding.
3. Validation Rules
  - Developed validation rules to maintain data integrity, ensuring accurate entries in the Lease object and preventing incomplete or incorrect submissions.
4. Email Templates
  - Created custom email templates to support tenant communications:
    - Tenant Leaving Notification
    - Leave Approved
    - Leave Rejection
    - Monthly Payment Reminder
    - Successful Payment Confirmation
5. Approval Process
  - Configured an approval process to check for vacant properties:
    - Initial Submission Action: Sends for review upon submission.

- Final Approval Action: Marks the property as leased and initiates tenant communication.
  - Final Rejection Action: Triggers notification to tenant with rejection details.
6. Apex Triggers and Classes
    - Developed an Apex Trigger to automate key lease management tasks.
    - Created an Apex handler class to manage specific functions within the system, including the validation of lease terms and calculation of payment due dates.
  7. Flows
    - Designed a Monthly Payment Flow to streamline payment processing for tenants, supporting reminders and status tracking.
  8. Scheduled Apex Class
    - Created an Apex class for scheduled operations, automating monthly billing reminders and payment checks to ensure timely notifications and compliance.

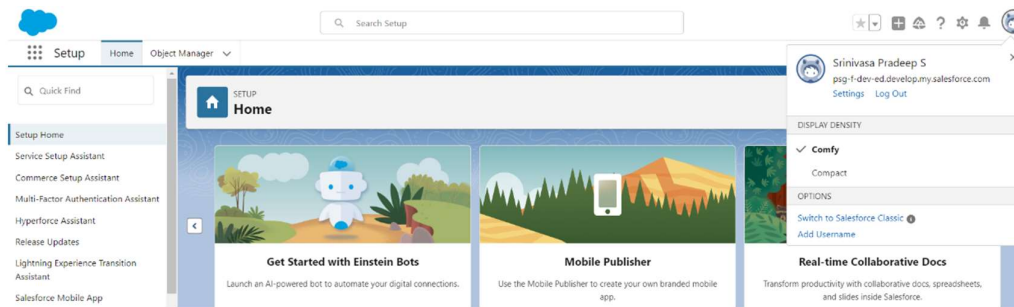
## 4. Detailed Steps to Solution Design

### 1. Created Salesforce Developer Account

- We began by creating a Salesforce Developer Account to access a development environment where we could build and customize the application. This setup provided the necessary tools to design, test, and deploy the application.

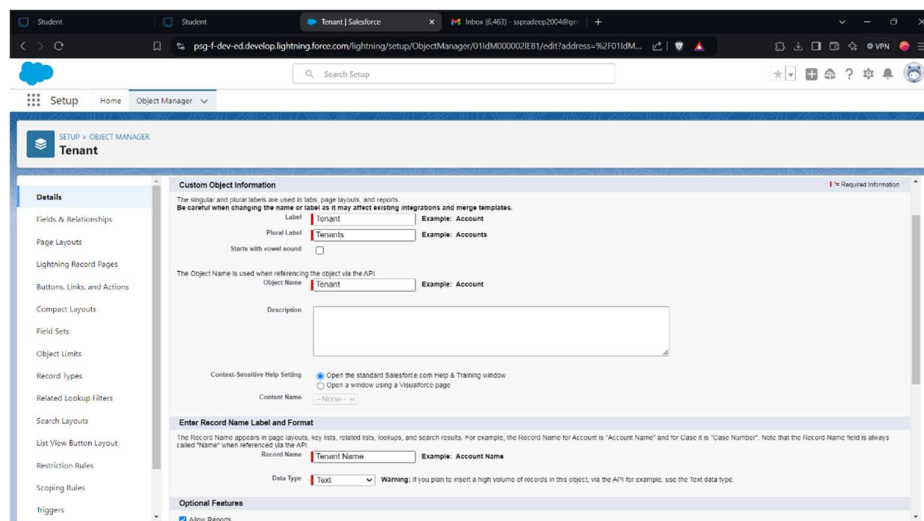
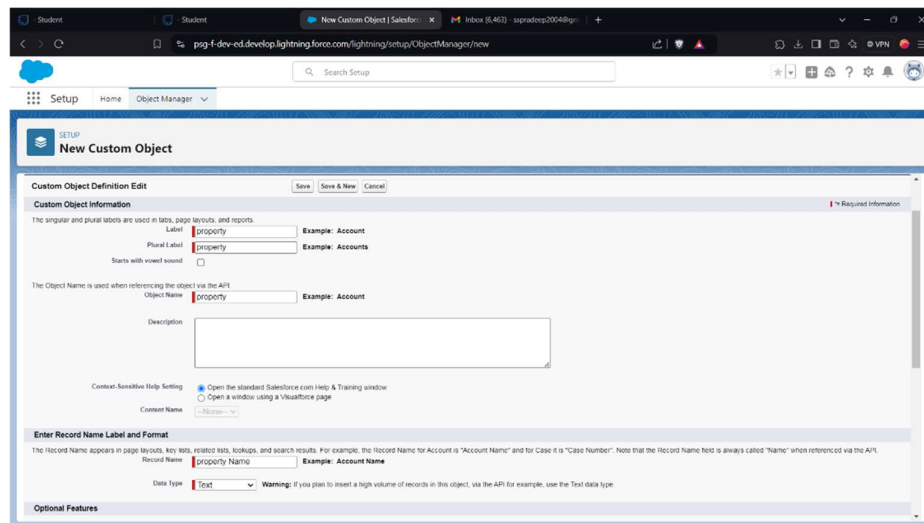


The screenshot shows the Salesforce Developer Edition sign-up page. On the left, there is a blue banner with the text "Build enterprise-quality apps fast to bring your ideas to life" and a list of bullet points: "Build apps fast with drag and drop tools", "Customize your data model with clicks", "Go further with Apex code", "Integrate with anything using powerful APIs", "Stay protected with enterprise grade security", and "Customize UI with clicks or any leading-edge web framework". On the right, there is a white sign-up form with the following fields: "First Name" (Srinivasa Pradeep), "Last Name" (S), "Email" (sspradeep2004@gmail.com), "Role" (Developer), "Company" (PSG), "Country/Region" (India), "State/Province" (Tamil Nadu), "Postal Code" (641062), and "Username" (srinivas@psg.com).



## 2. Object Definitions

- Defined primary objects: **Property**, **Tenant**, **Lease**, and **Payment** within Object Manager.
- Each object was designed to store specific data: Property details, tenant information, lease agreements, and payment records.



**SETUP > OBJECT MANAGER**  
**lease**

**Details**

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

**Custom Object Information**

The singular and plural labels are used in tabs, page layouts, and reports. Be careful when changing the name or label as it may affect existing integrations and merge templates.

Label:  Example: Account  
Plural Label:  Example: Accounts  
Starts with vowel sound: ☐

The Object Name is used when referencing the object via the API.  
Object Name:  Example: Account

Description:

Context Sensitive Help Setting: ☒ Open the standard Salesforce.com Help & Training window  
☐ Open a window using a Visualforce page

Content Name:

**Enter Record Name Label and Format**

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name:  Example: Account Name  
Data Type:  Warning: if you plan to insert a high volume of records in this object, via the API for example, use the Text data type.

**Optional Features**

☒ Allow Reports

**SETUP > NEW CUSTOM OBJECT**  
**New Custom Object**

**Custom Object Information**

The singular and plural labels are used in tabs, page layouts, and reports.

Label:  Example: Account  
Plural Label:  Example: Accounts  
Starts with vowel sound: ☐

The Object Name is used when referencing the object via the API.  
Object Name:  Example: Account

Description:

Context Sensitive Help Setting: ☒ Open the standard Salesforce.com Help & Training window  
☐ Open a window using a Visualforce page

Content Name:

**Enter Record Name Label and Format**

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name:  Example: Account Name  
Data Type:  Warning: if you plan to insert a high volume of records in this object, via the API for example, use the Text data type.

**Optional Features**

☒ Allow Reports  
☒ Allow Activities

### 3. Tab Configuration

- Configured tabs for each main object to facilitate quick access.
- Enabled tabs for Property, Tenant, Lease, and Payment, allowing users to manage lease data seamlessly.

**SETUP > TABS**  
**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.

**Custom Object Tabs**

Action	Label	Tab Style	Description
EDIT   DEL	Enhancement Requests	Building Block	
EDIT   DEL	lease	Blank	
EDIT   DEL	Payment	Blank	
EDIT   DEL	property	Building	
EDIT   DEL	Research Proposal	Signature	
EDIT   DEL	Tenant	People	

**Web Tabs**

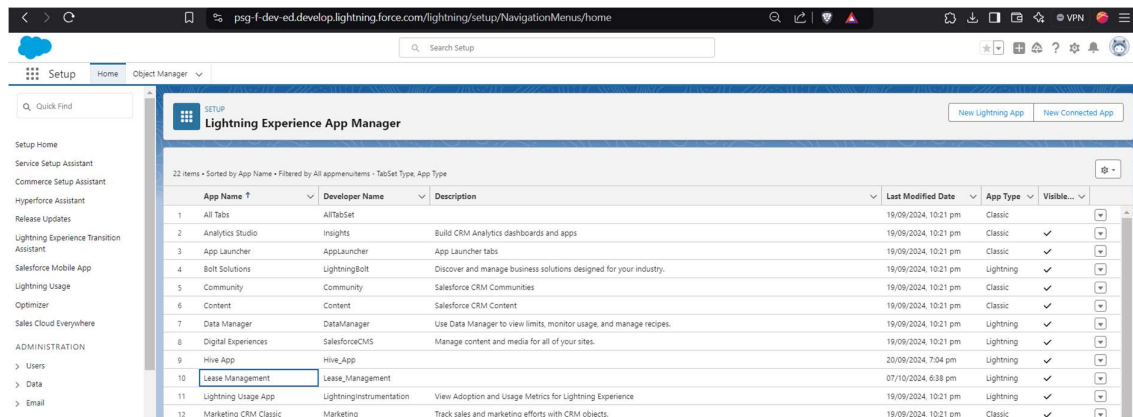
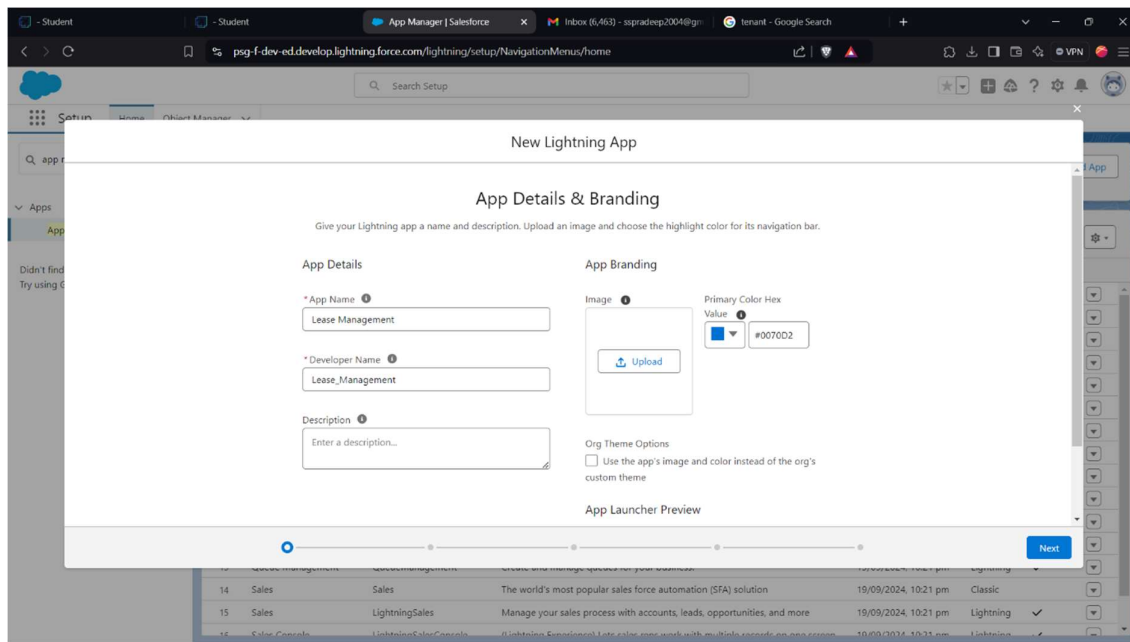
No Web Tabs have been defined.

**Visualforce Tabs**

No Visualforce Tabs have been defined.

## 4. Lightning App Development

- Created a custom Lightning App named **Lease Management** to consolidate all functionalities.
- Configured branding, navigation items, and access settings to improve user experience and accessibility.



## 5. Custom Field Creation

- Created custom fields within each object to capture required details:
  - **Property:** Location, type, rental amount, availability status.

property | Salesforce

psg-f-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01dM000002IE6P/FieldsAndRelationships/v...

Setup Home Object Manager

property

Details

Fields & Relationships

8 Items. Sorted by Field Label

Quick Find New Deleted Fields Field Dependencies Set History Tracking

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address__c	Long Text Area(32768)		
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Name	Name__c	Text(25)		
Owner	OwnerId	Lookup(User,Group)		✓
property Name	Name	Text(80)		✓
sftt	sftt__c	Text(18)		
Type	Type__c	Picklist		

- **Tenant:** Contact information, lease start/end dates.

Tenant | Salesforce

psg-f-dev-ed.develop.lightning.force.com/lightning/setup/ObjectManager/01dJM000002IE81/FieldsAndRelationships/v...

Setup Home Object Manager

Tenant

Details

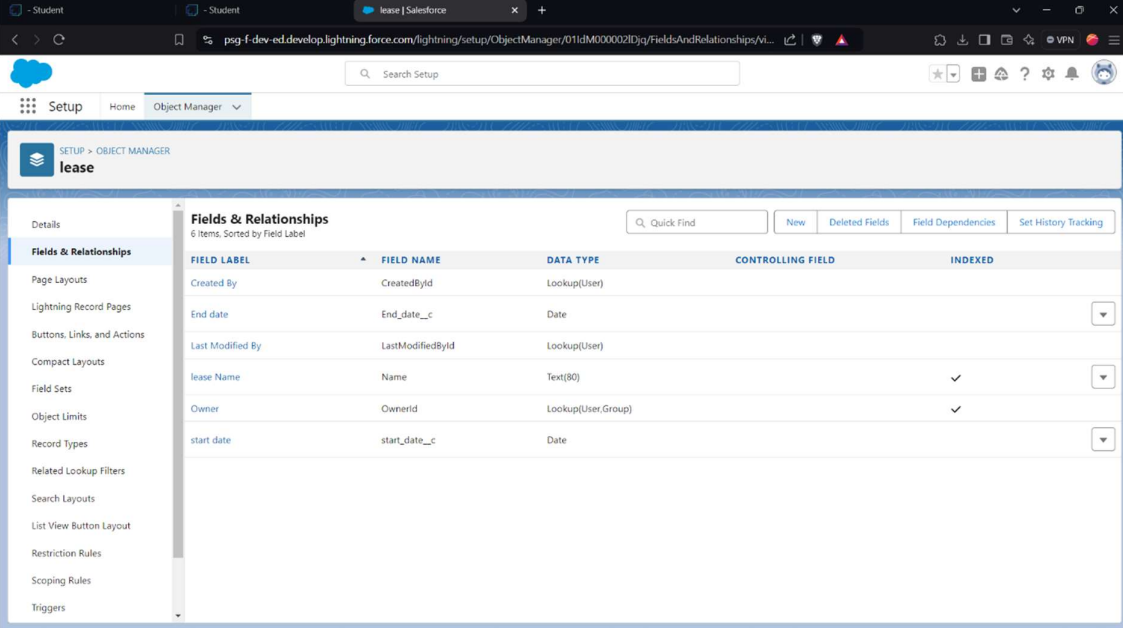
Fields & Relationships

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)		
Owner	OwnerId	Lookup(User,Group)		✓
Phone	Phone__c	Phone		
status	status__c	Picklist		
Tenant Name	Name	Text(80)		✓

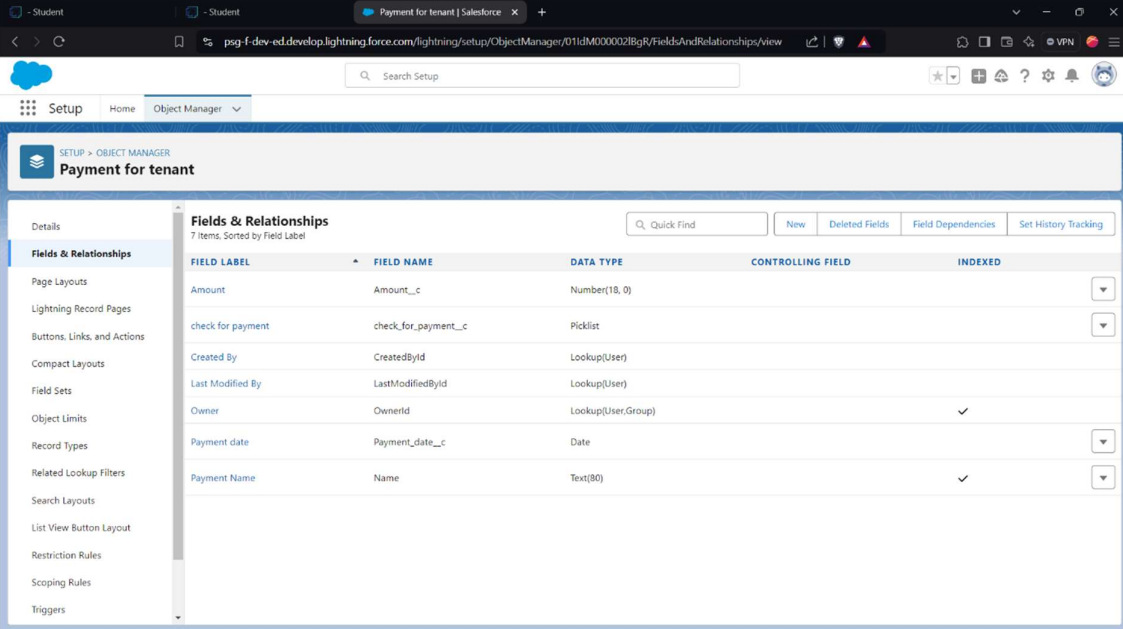
- **Lease:** Lease terms, conditions, and expiration date.



The screenshot shows the Salesforce Object Manager interface for the 'Lease' object. The left sidebar contains a navigation menu with options like Details, Fields & Relationships, Page Layouts, and others. The main area displays the 'Fields & Relationships' section for the 'Lease' object, showing 6 items sorted by Field Label. The table below lists the fields and their properties.

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)		✓
start date	start_date__c	Date		

- **Payment:** Amount due, due date, and payment status.



The screenshot shows the Salesforce Object Manager interface for the 'Payment for tenant' object. The left sidebar contains a navigation menu with options like Details, Fields & Relationships, Page Layouts, and others. The main area displays the 'Fields & Relationships' section for the 'Payment for tenant' object, showing 7 items sorted by Field Label. The table below lists the fields and their properties.

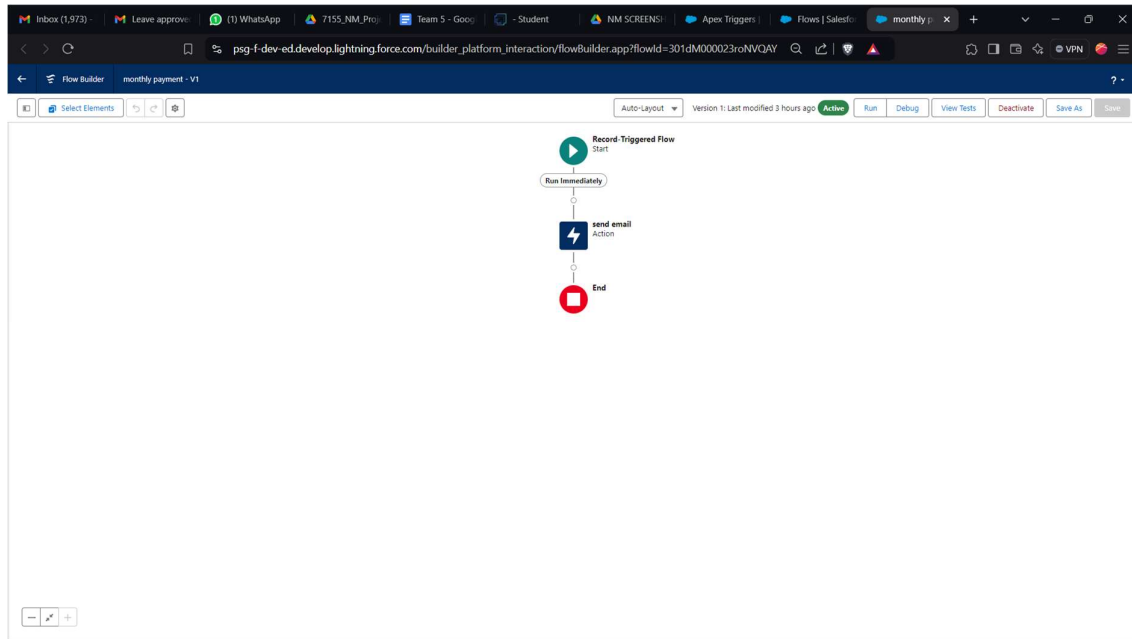
FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Amount	Amount__c	Number(18, 0)		
check for payment	check_for_payment__c	Picklist		
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Payment date	Payment_date__c	Date		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

## 6. Flow Creation for Monthly Payments

- Developed a **Monthly Payment Flow** to guide users through the payment process for tenants.

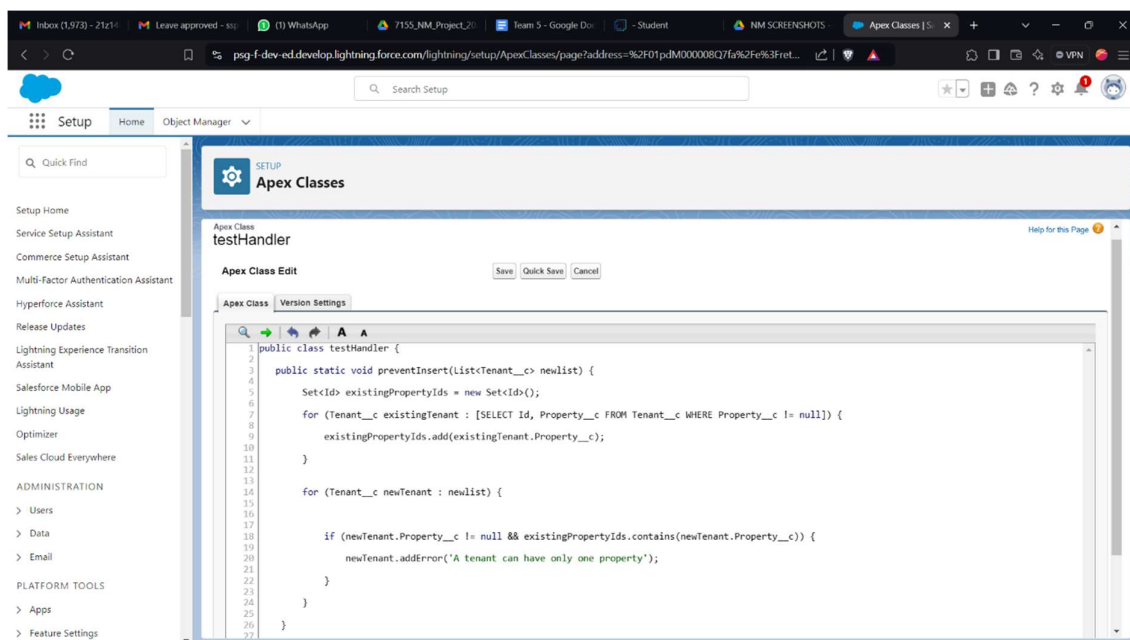


- Included automated reminders and status tracking, streamlining monthly rent collection.



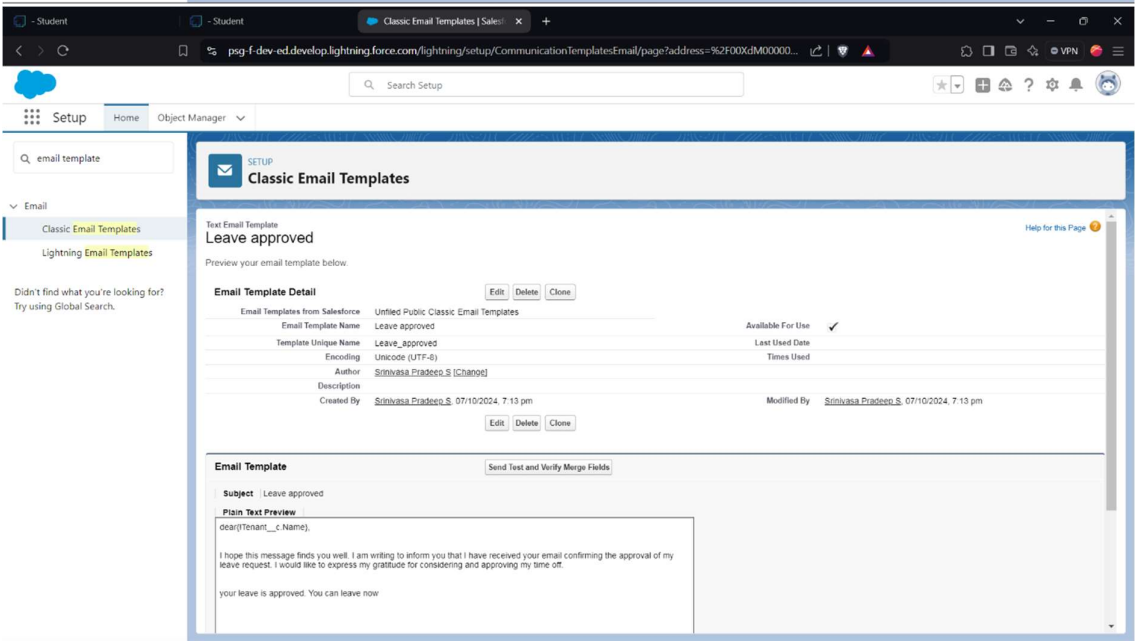
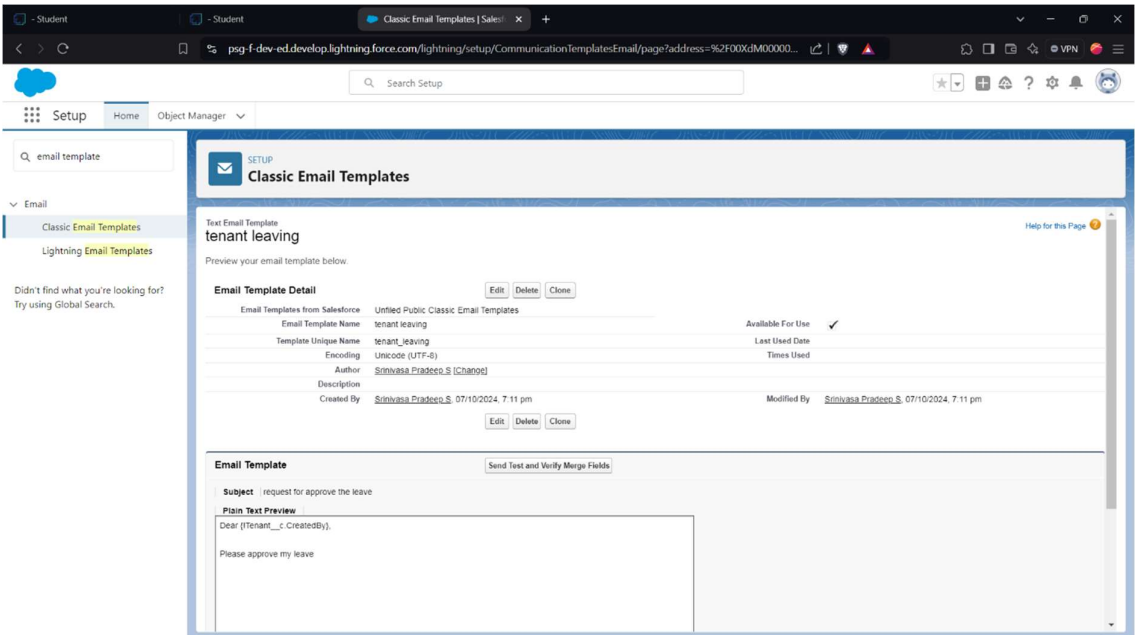
## 7. Apex Trigger and Handler Class

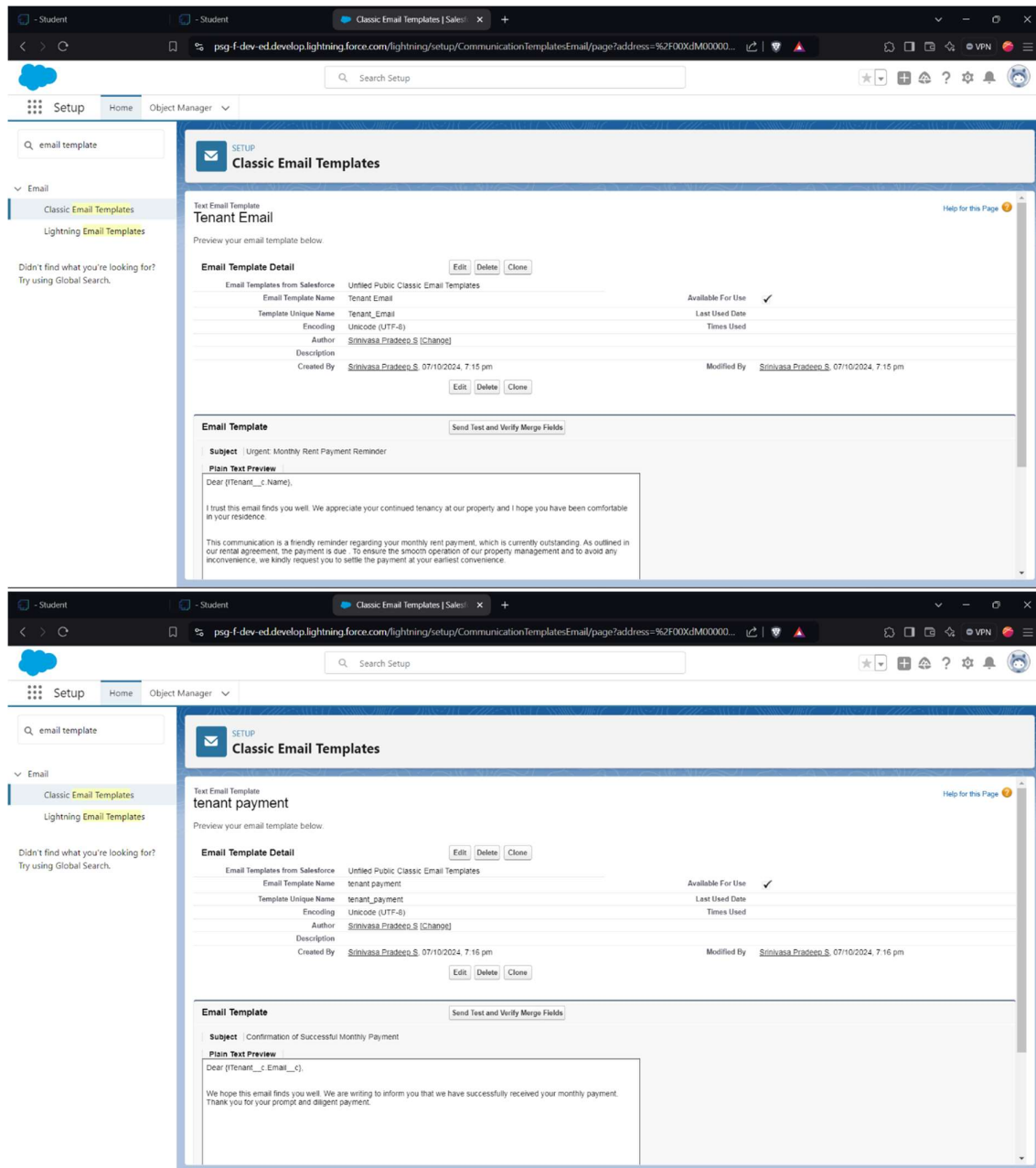
- Created an Apex Trigger to automate lease calculations, such as due dates and reminders.
- Developed an Apex handler class to process complex calculations and manage lease status changes.



## 8. Email Template Setup

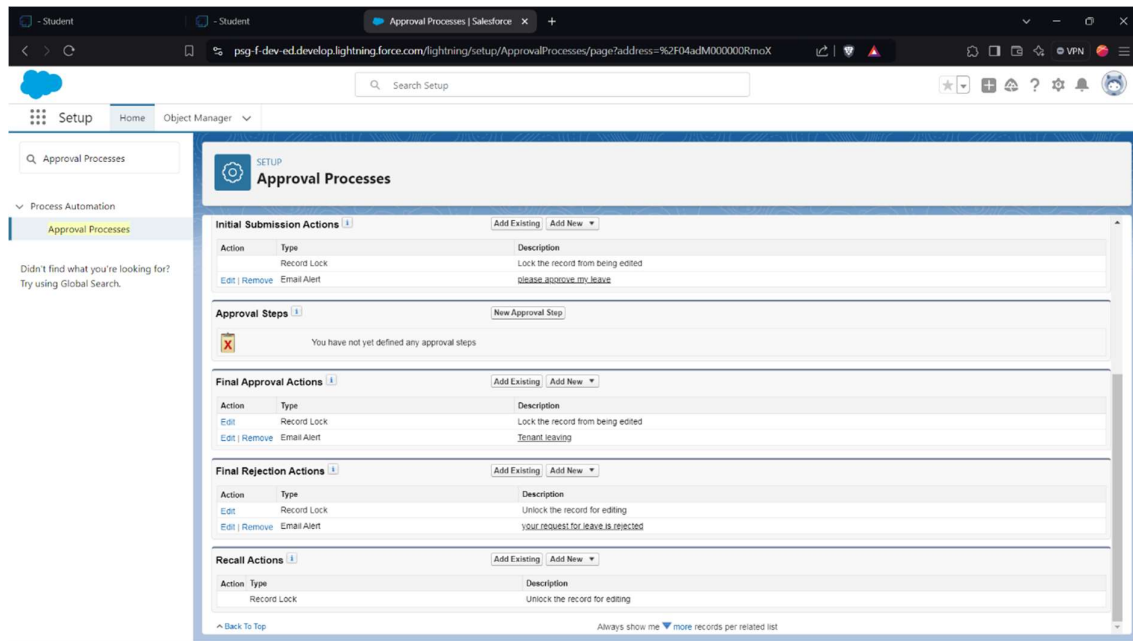
- Designed email templates for tenant communication, including notifications for payment reminders, successful payments, and lease approval or rejection.





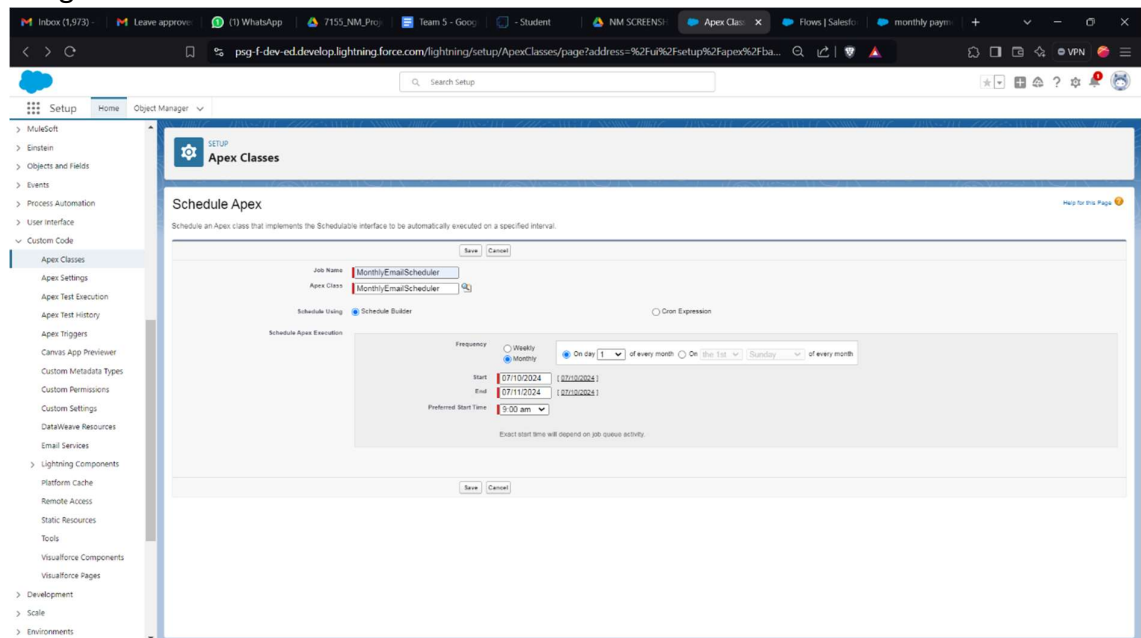
## 9. Approval Process for Vacant Properties

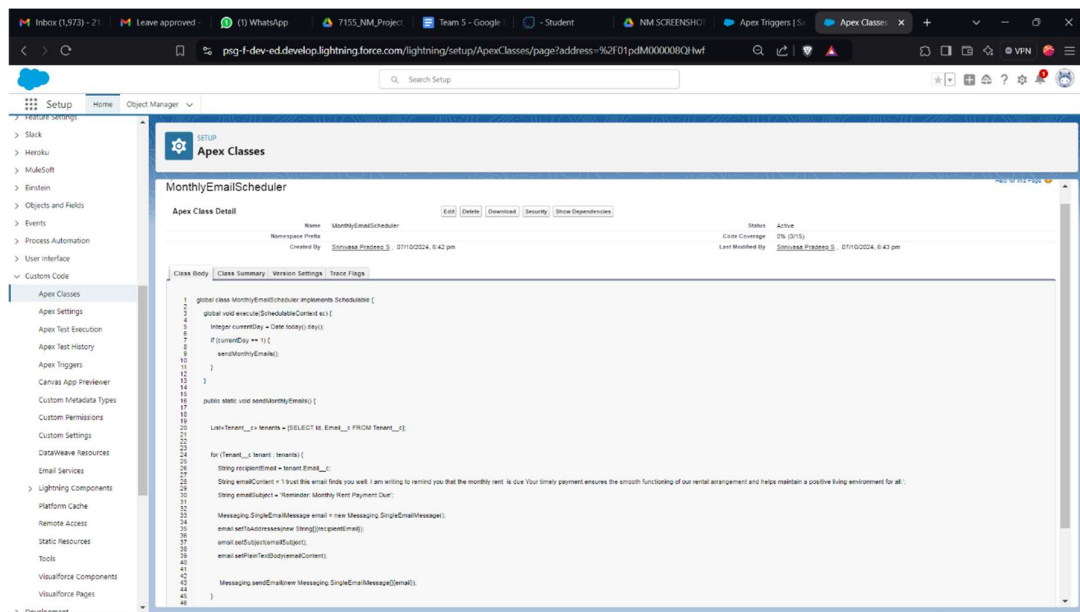
- Established an approval process to verify property availability before leasing:
  - Configured initial submission, final approval, and rejection actions to streamline the lease request process.



## 10. Scheduled Apex Class for Monthly Reminders

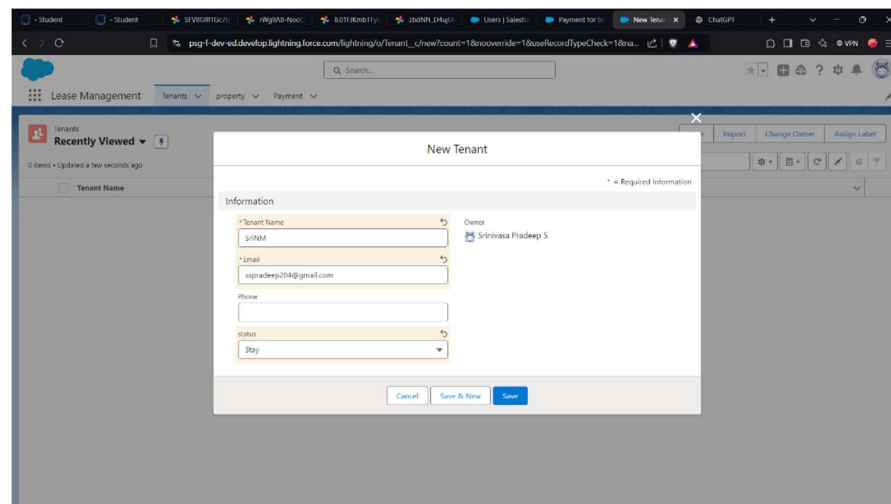
- Scheduled an Apex class to handle monthly billing reminders and lease renewals, ensuring tenants and administrators are up-to-date on upcoming obligations.





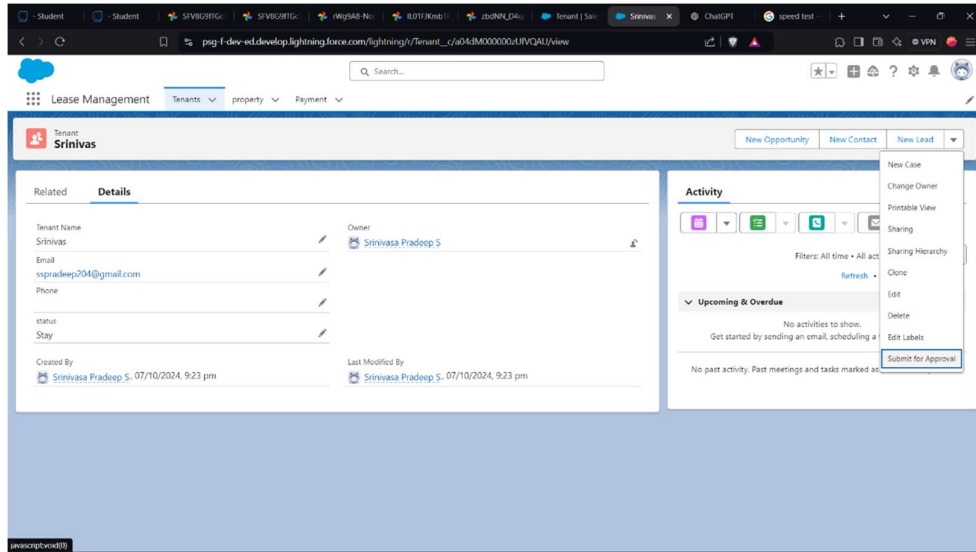
## 11. Create a Tenant Record

- Created a **Tenant** record within the system, entering all relevant details such as contact information, lease start and end dates, and tenant status.
- Ensured that the new tenant record is associated with the appropriate **Property** and **Lease** records through lookup relationships, enabling seamless data access and association.



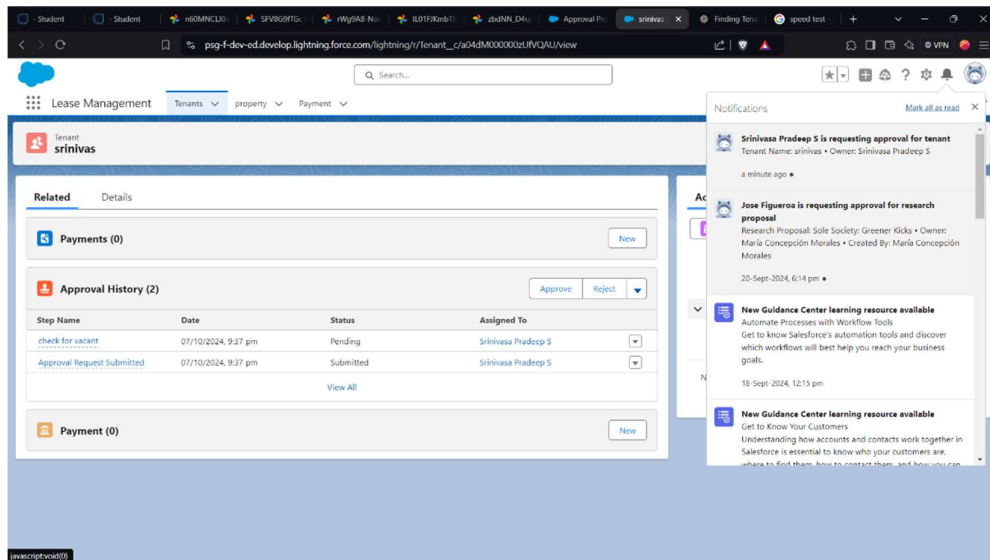
## 12. Submit for Approval

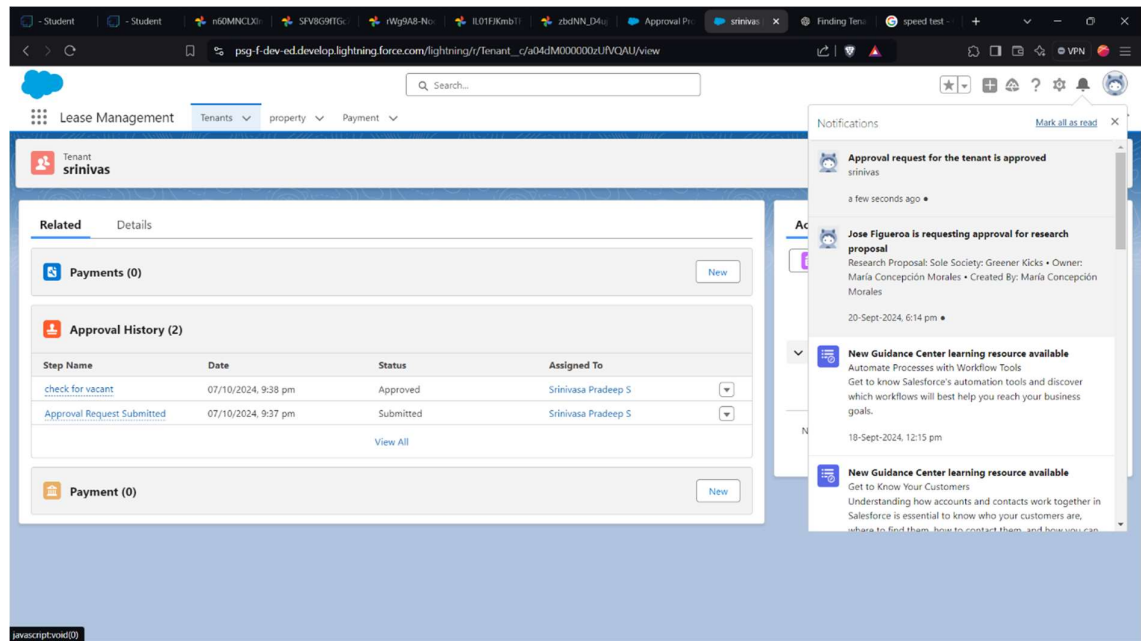
- Initiated the **Approval Process** for the new lease by submitting the **Lease** record for approval.
- The submission triggers an initial review, allowing the leasing team to validate all provided information before final approval.



### 13. Wait for Approval

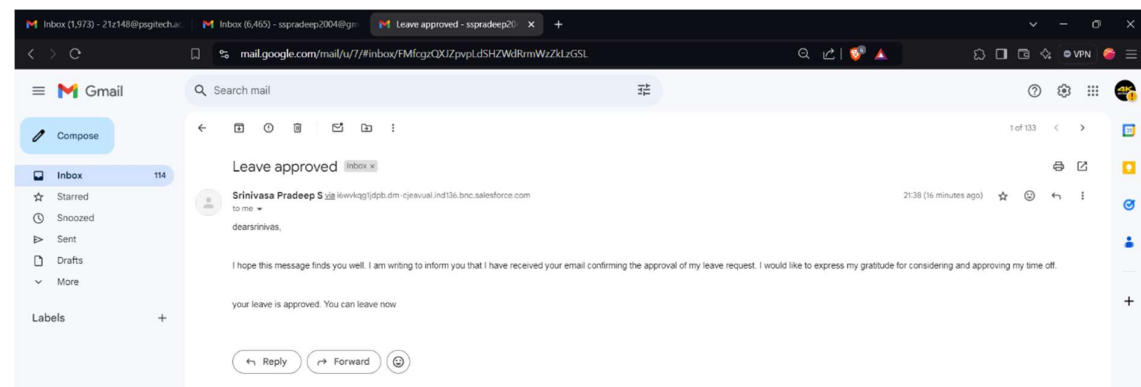
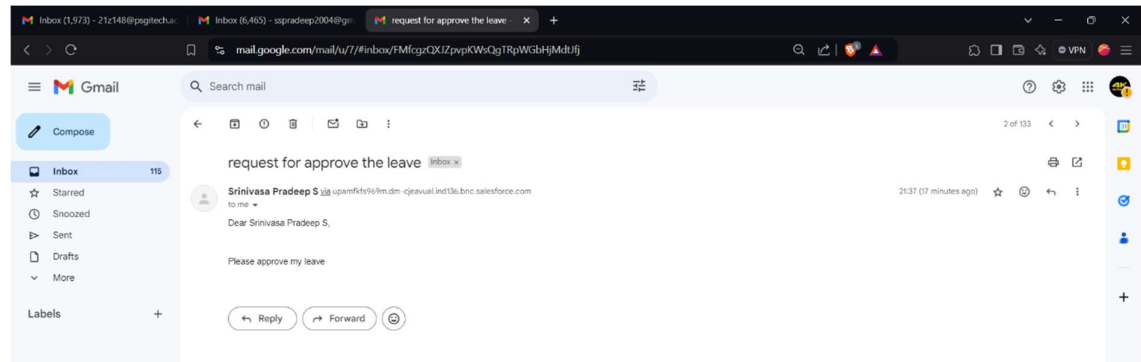
- The **Approval Process** waits for the approver's action (approval or rejection). During this phase, the lease status remains in a "Pending Approval" state until final action is taken.





#### 14. Email Sent for Approval Confirmation

- Upon approval, an automated **Email Template** for lease approval is triggered and sent to the tenant, confirming the lease's approval status.
- The email outlines the lease's terms and conditions, ensuring clear communication with the tenant.



## 5. Testing and Validation

### Unit Testing (Apex Classes, Triggers)

#### Apex Trigger:

trigger test on Tenant\_\_c (before insert)

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

#### Test 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');  
            }  
        }  
    }  
}
```

#### MonthlyEmailScheduler:

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});

    }

}

```

## Steps:

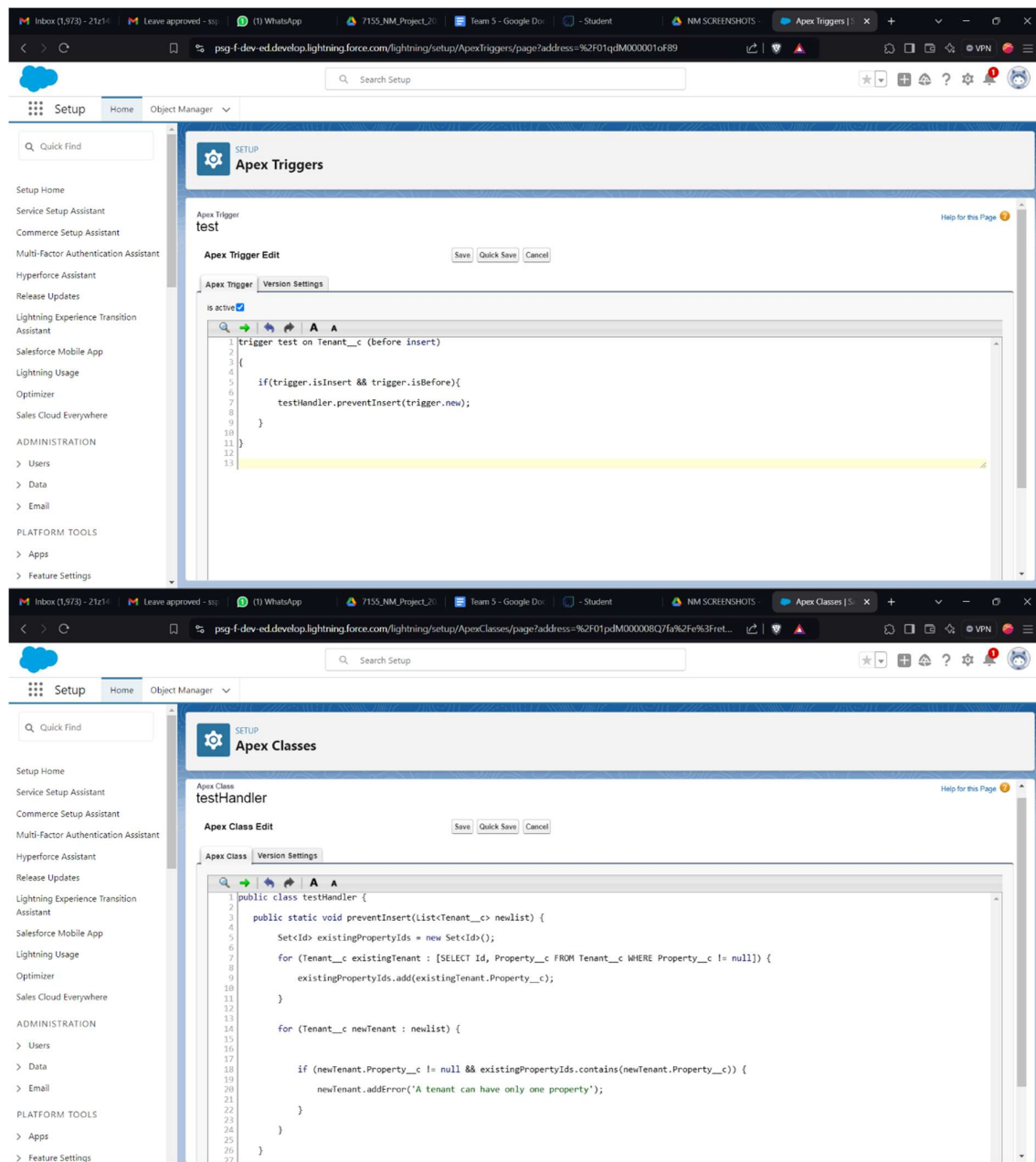
1. Created a Test Property Record: Created a Property\_\_c record to serve as the property for assigning tenants.
2. Inserted the First Tenant: Inserted a Tenant\_\_c record linked to the test property, simulating the initial tenant assignment.
3. Prepared a Second Tenant for the Same Property: Created another Tenant\_\_c record with the same property ID to test the trigger's restriction on multiple tenants per property.
4. Inserted the Second Tenant: Attempted to insert the second tenant within a try-catch block to catch the expected error, confirming that the trigger prevented the insertion.

5. Verified Error Message: Checked if the error message matched the expected output: "A tenant can have only one property," verifying that the trigger works as intended.

6. Run Apex Test Execution:

- Go to Setup > Apex Test Execution.
- Select TenantTriggerTest and click Run.

Confirm that all assertions pass, indicating that the trigger prevents multiple tenants from being assigned to the same property.



Logs							
User	Application	Operation	Time	Status	Read	Size	
Srinivasa Pradeep S	Unknown	ApexTestHandler	10/7/2024, 11:56:46 PM	Success	Unread	5.49 KB	
Srinivasa Pradeep S	Unknown	ApexTestHandler	10/7/2024, 11:56:43 PM	Success	Unread	485 bytes	
Srinivasa Pradeep S	Unknown	ApexTestHandler	10/7/2024, 11:56:43 PM	Success	Unread	2.18 KB	
Srinivasa Pradeep S	Unknown	ApexTestHandler	10/7/2024, 11:56:43 PM	Success	Unread	18.06 KB	
Srinivasa Pradeep S	Unknown	ApexTestHandler	10/7/2024, 11:56:43 PM	Success	Unread	1.48 KB	
Srinivasa Pradeep S	Unknown	ApexTestHandler	10/7/2024, 11:56:43 PM	Success	Unread	23.35 KB	

## 6. Conclusion

**Summary of Achievements:** The Lease Management project successfully established an efficient, transparent, and scalable system for managing property leases. Key accomplishments include:

- **Organized Data Management:** Created custom objects, fields, and tabs for structured lease tracking.
- **Automation of Lease Processes:** Implemented triggers, validation rules, and scheduled classes to streamline recurring processes like monthly payments and notifications.
- **Improved Communication:** Developed approval processes and email templates for automated tenant communication, improving clarity and tenant experience.
- **Enhanced Decision-Making:** Built custom reports and dashboards to provide stakeholders with real-time insights, facilitating faster, more informed decisions.

This project illustrates the effective use of Salesforce in supporting lease management processes, driving operational efficiency, and ensuring data accuracy and compliance. Through this system, the lease management workflow has been optimized for better tracking and tenant relations.