

NAAN MUDHAVALVAN- SALESFORCE REPORT

A CRM Application To Manage The Mall

PROJECT CREATED BY
B.E –VII SEMESTER

SANTHIYA. B	712221104018
SRUTHIKKA.S	712221104022
SHREENITHI. S	712221104020
MEGHA.M.R.	712221104005

DEPARTMENT OF COMPUTER SCIENCE
PARK COLLEGE OF ENGINEERING AND TECHNOLOGY

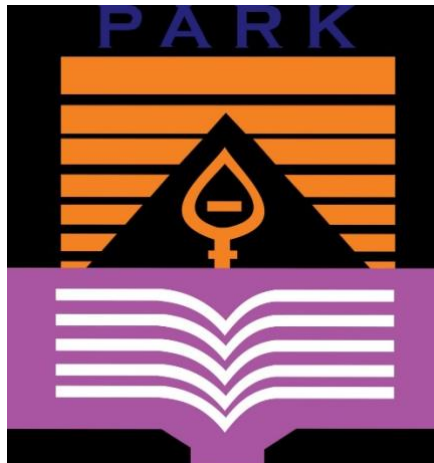


Table of Contents

1.Project Overview.....	3
2.Objectives	3
Busines Goals	3
Specific Outcomes	4
3. Salesforce Key Features and Concepts Utilized	4
Creating Custom Objects.....	4
Automation	4
Reports and Dashboards.....	4
Asynchronous Apex.....	4
Custom Validation Rules.....	4
4.Detailed Steps to Solution Design.....	5
Why Are We Using Salesforce Platform for This Project?	5
Creating developer account.....	5
Creating cutom objects.....	6
Create the Lightning App.....	9
Steps to Create Custom App in Salesforce.....	9
Record Insertion	10
Create flows.....	12
Apex Triggers.....	15
Asynchronous Apex.....	19
Create Reports and Dashboards	21
5. Testing and Validation.....	25
6. Key Scenarios Addressed by Salesforce in the Implementation Project	
7.Conclusion.....	26

1. Project Overview

The Management App is a dynamic and scalable CRM system designed specifically for managing commercial malls. Built on the Salesforce platform, the application offers solutions to the challenges faced by mall administrators, including managing tenant information, tracking leases, and resolving tenant issues efficiently.

Additional Features of the Management App

- **Scalable Architecture:** The app can be expanded to handle multiple malls and thousands of tenant records without performance degradation.
- **Mobile Access:** Fully accessible via Salesforce mobile apps, ensuring on-the-go usability for mall administrators and staff.
- **Customizable Analytics:** Personalized dashboards and reports can be tailored to specific management needs.
- The app addresses both day-to-day operations and long-term planning requirements, making it a comprehensive tool for mall administrators.

2. Objectives

The objectives of the Management App are both business-oriented and functionality-driven:

- **Business Goals:**
 - **Enhanced Operational Efficiency:** Automate and streamline lease tracking, tenant communication, and data management processes.
 - **Improved Tenant Relationship Management:** Foster a more proactive, organized approach to handling tenant concerns, lease inquiries, and contract management.
 - **Strategic Planning and Insights:** Utilize Salesforce analytics to gain insights into occupancy trends, tenant performance, and potential growth opportunities within the mall

- **Specific Outcomes:**
 - **Accurate Lease Tracking:** Automate reminders for lease renewals and terminations, reducing the risk of lost revenue or lease lapses.
 - **Enhanced Communication:** Centralize communication with tenants to address inquiries and resolve issues quickly.
 - **Informed Decision Making:** Implement real-time dashboards for metrics like occupancy rates, revenue projections, and tenant feedback.

3. Salesforce Key Features and Concepts Utilized

- **Custom Objects:** Lease, Tenant, and Tenant Issues for data management.
- **Automations:** Workflows, Apex triggers, and flows to streamline repetitive tasks.
- **Reports and Dashboards:** Insights on lease statuses, tenant issues, and financial summaries.
- **Asynchronous Apex:** For scheduled tasks such as periodic updates and batch processing.
- **Email Templates:** Configured pre-defined templates for automated tenant communications, such as reminders for overdue rent or service updates.
- **Chatter Integration:** Enabled internal collaboration by integrating Salesforce Chatter for real-time updates and discussions on tenant-related issues.
- **Audit Trails:** Leveraged Salesforce's tracking mechanisms to monitor changes to key records, enhancing accountability.
- **Custom Validation Rules:** Implemented rules to ensure data integrity, such as validating tenant PAN numbers or mandatory fields during lease creation.

The Salesforce ecosystem provides the flexibility to adapt to the mall's specific operational needs while ensuring a user-friendly interface for all stakeholders.

4. Detailed Steps to Solution Design

CRM Application to Manage the Mall

This section outlines the structured approach to building the CRM application step-by-step.

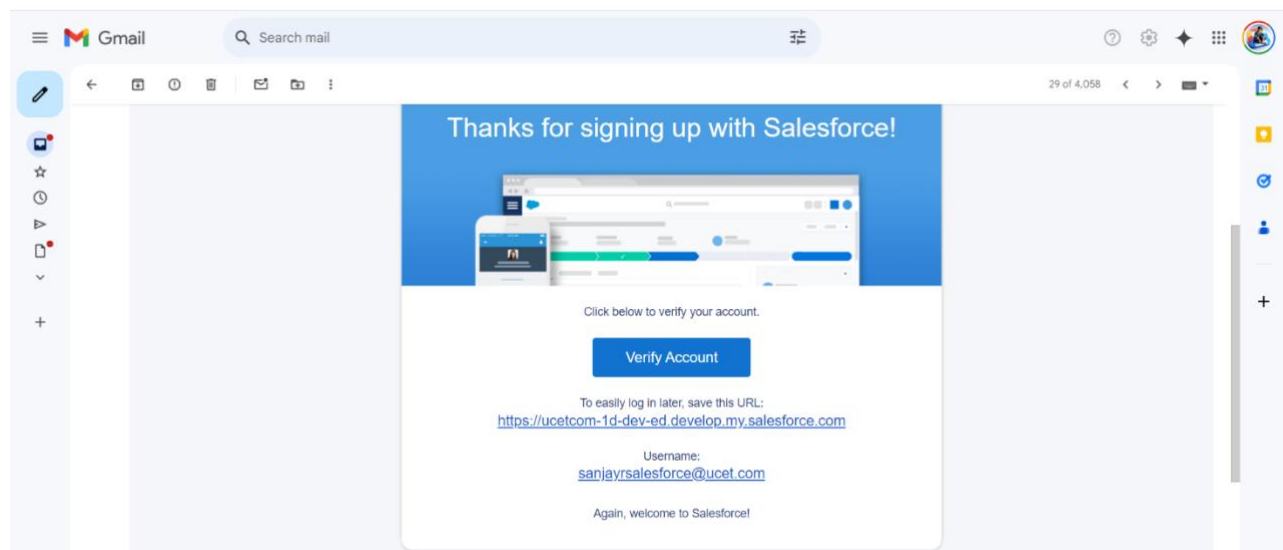
1. Why Are We Using Salesforce Platform for This Project?

Salesforce offers a robust, secure, and scalable platform for developing CRM applications with minimal code and extensive customization options. Features like Lightning UI, workflows, automation tools, and advanced reporting make Salesforce ideal for managing a mall's operations.

2. Creating Developer Account

- A Salesforce developer account is created to provide access to the Salesforce Lightning platform and tools needed for custom app development.

Figure:1

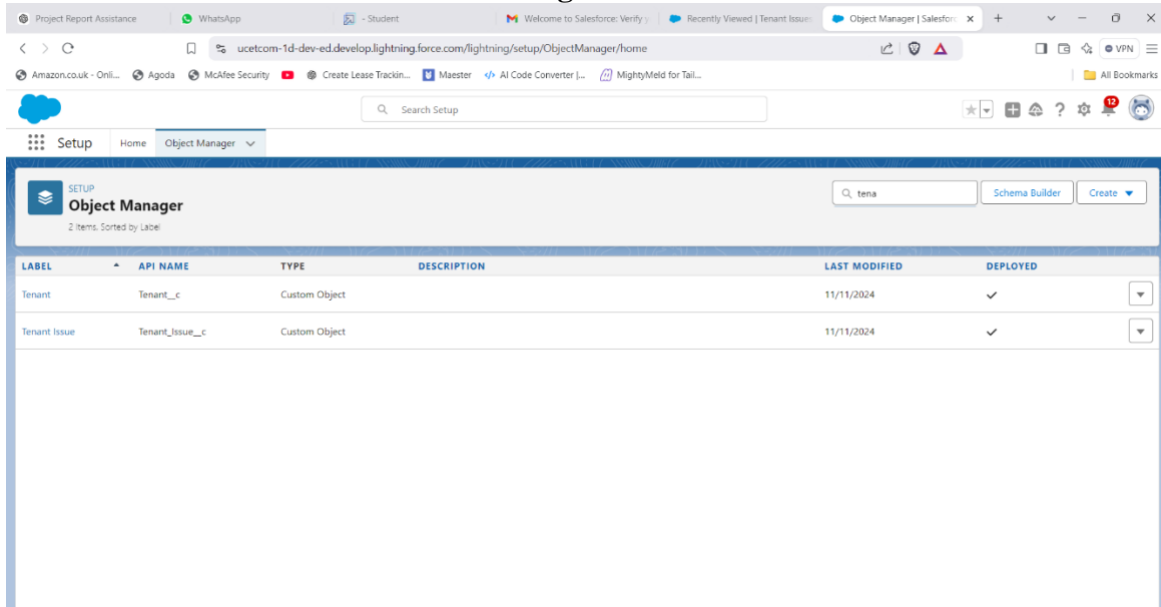


3. Create Custom Objects

Custom objects are created to manage mall-specific data:

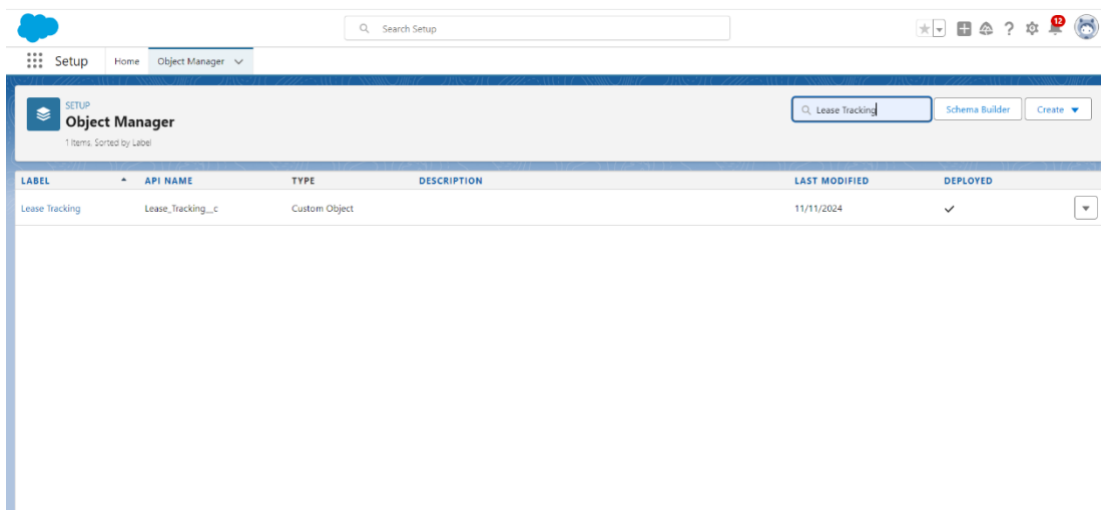
- **Tenant Object:** For storing tenant details like contact info, lease dates, and PAN card details

Figure:2



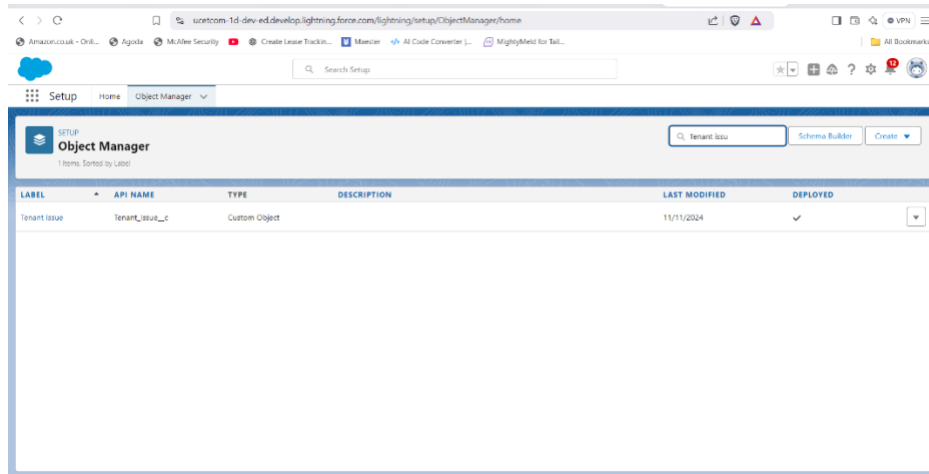
- **Lease Tracking Object:** For tracking lease agreements, including payment statuses.

Figure:3



- **Tenant Issues Object:** For logging tenant complaints, service requests, and maintenance records.

Figure:4



4. Create Custom Tabs for Tenant Object

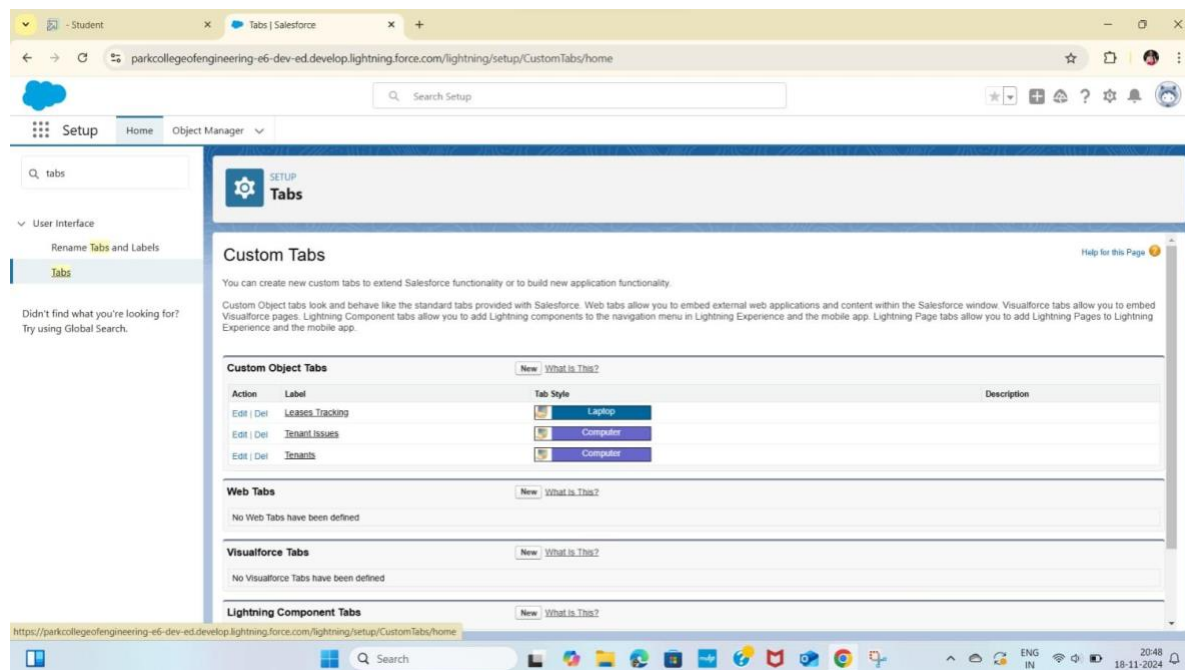


Figure:5

- Custom tabs are designed to allow easy access to the Tenant object through the

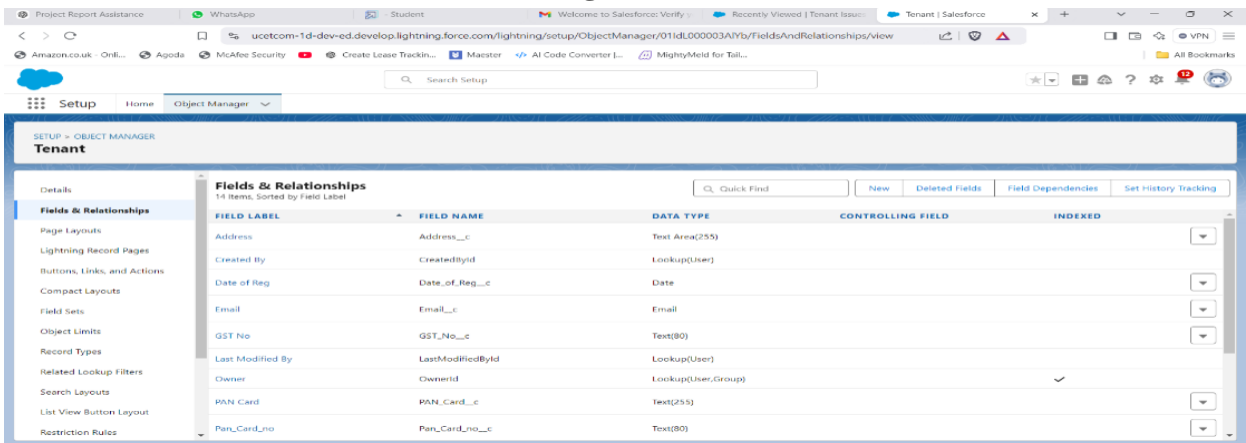
Salesforce UI.

5. Create Fields and Relationships

Fields are created to capture necessary data and relationships are established between objects to ensure seamless data interaction:

- **Tenant Object Fields:** Tenant Name, Contact Details, PAN Number, Lease ID (lookup).

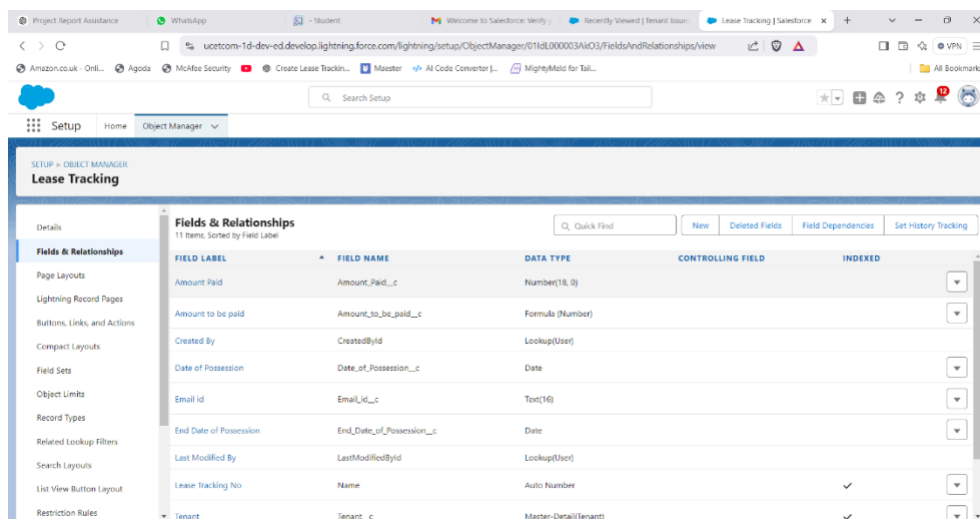
Figure:6



FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address__c	Text Area(255)		
Created By	CreatedById	Lookup(User)		
Date of Reg	Date_of_Reg__c	Date		
Email	Email__c	Email		
GST No	GST_No__c	Text(80)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
PAN Card	PAN_Card__c	Text(255)		
Pen_Card_No	Pen_Card_No__c	Text(80)		

- **Lease Tracking Fields:** Lease Start Date, End Date, Total Rent, Payment Status.

Figure:7



FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Amount Paid	Amount_Paid__c	Number(18, 0)		
Amount to be paid	Amount_to_be_paid__c	Formula (Number)		
Created By	CreatedById	Lookup(User)		
Date of Possession	Date_of_Possession__c	Date		
Email id	Email_Id__c	Text(16)		
End Date of Possession	End_Date_of_Possession__c	Date		
Last Modified By	LastModifiedById	Lookup(User)		
Lease Tracking No	Name	Auto Number		✓
Tenant	Tenant__c	Master-Detail(Tenant)		✓

- **Tenant Issues Fields:** Issue ID, Description, Status, Assigned To,

6. Create the Lightning App

- The Lightning app is designed to offer an intuitive interface with easy navigation between Tenant, Lease Tracking, and Tenant Issues data.

7. Steps to Create Custom App in Salesforce

- Navigate to the App Builder and create a custom app with navigation items for custom objects, dashboards, and reports. Additionally, I inserted a logo for smart mall app.

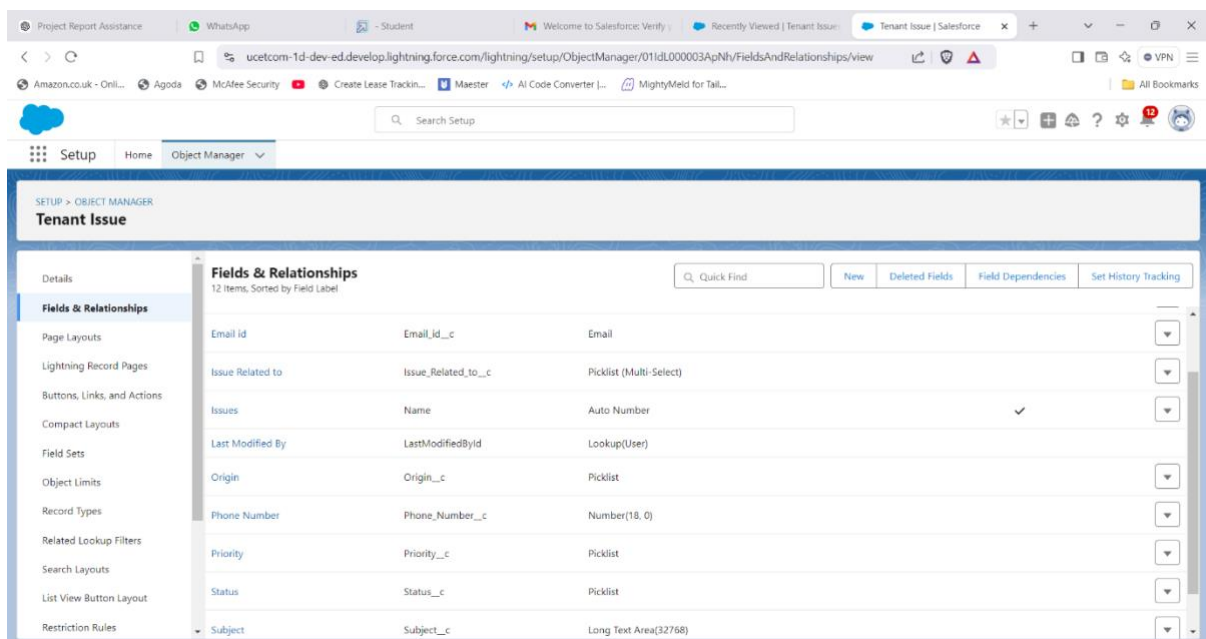
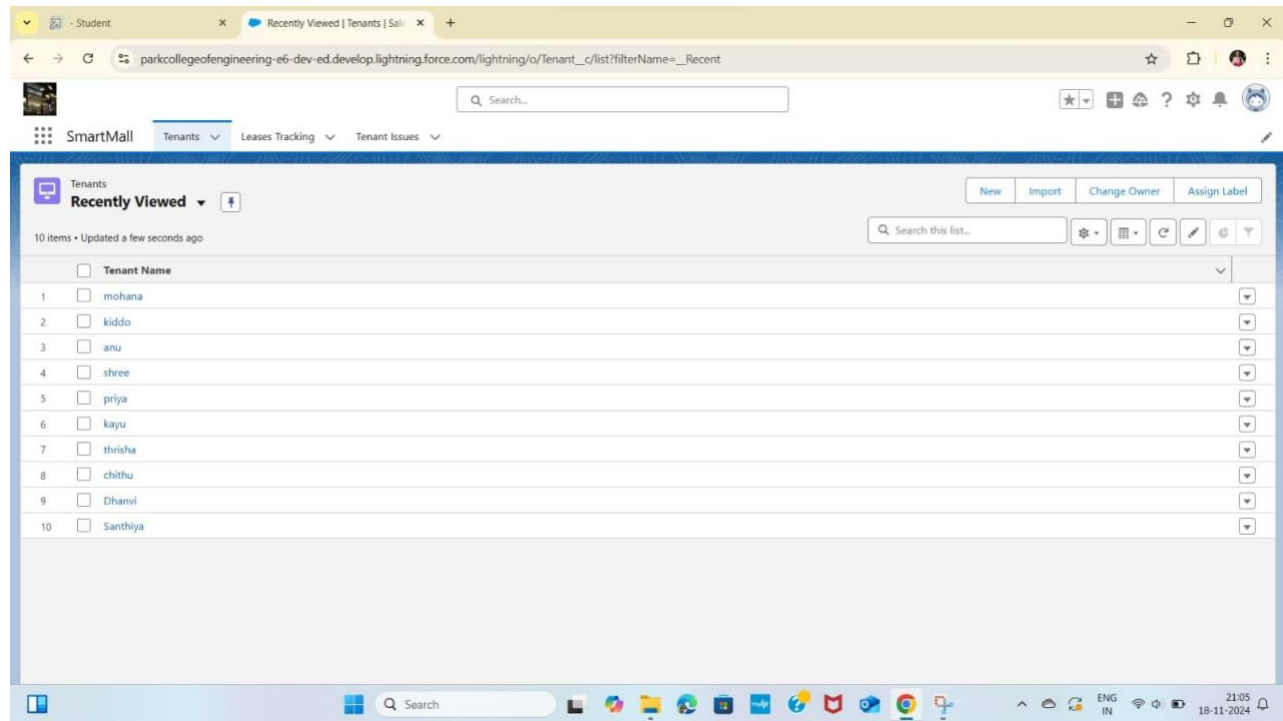


Figure:8

Figure:9



8. Record Insertion

Records are inserted for testing and demonstration purposes:

- **Tenant Object:** Add tenant details manually or through data import tools.

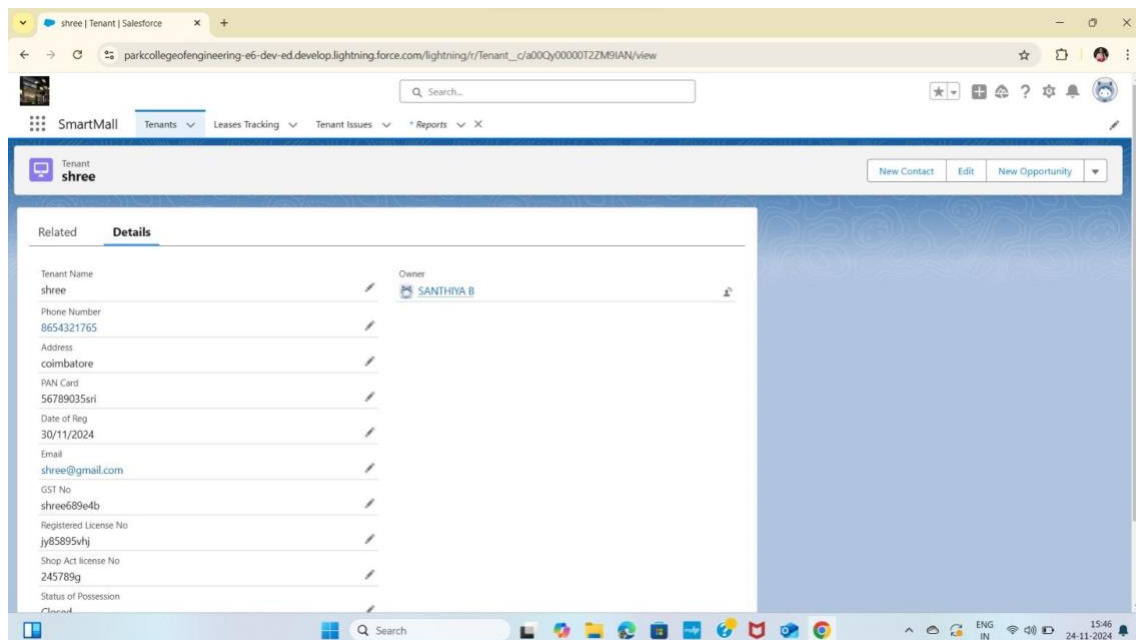


Figure:10

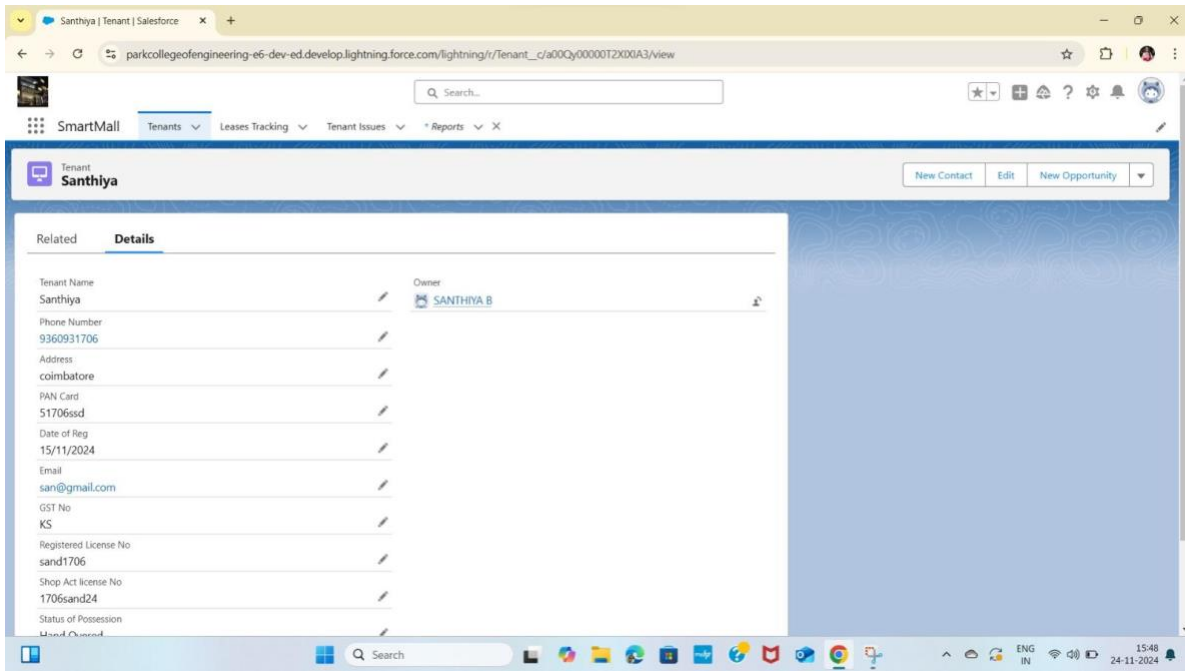
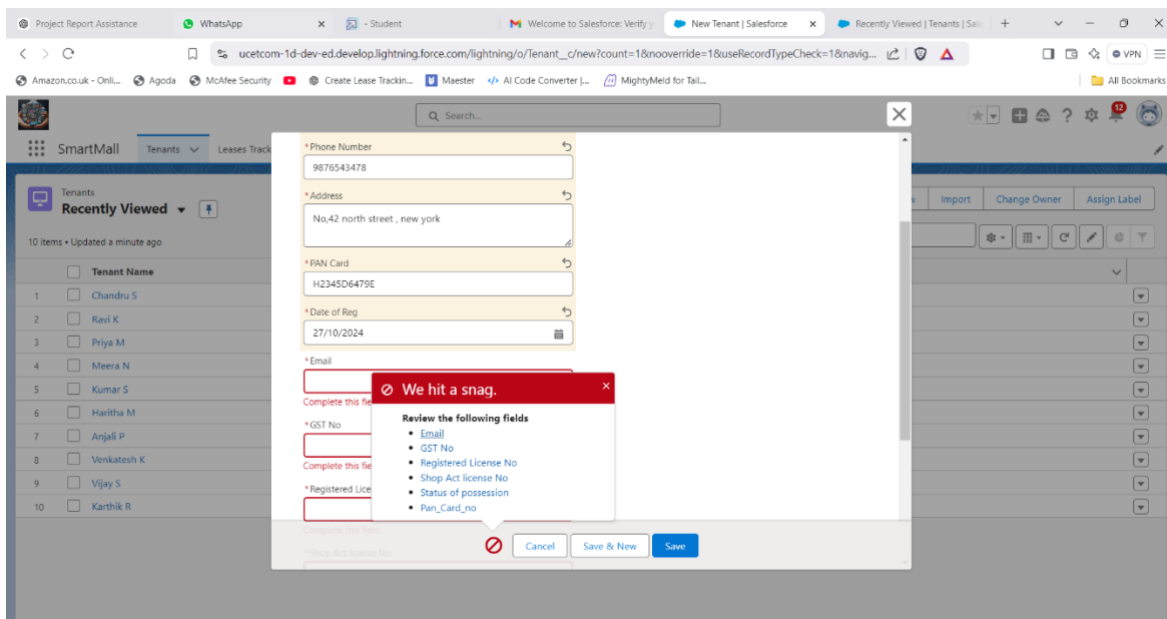


Figure:11

- **Lease Tracking Object:** Add lease agreement details linked to tenants.

Figure:12



- **Tenant Issues Object:** Log service requests or complaints raised by tenants.

Figure:13

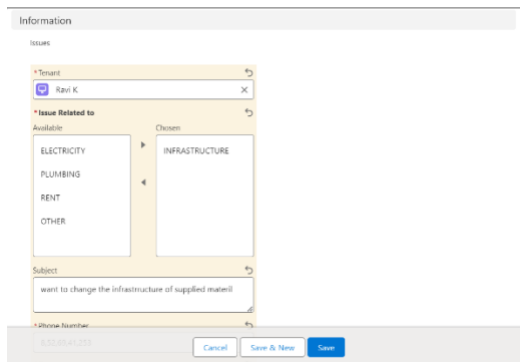
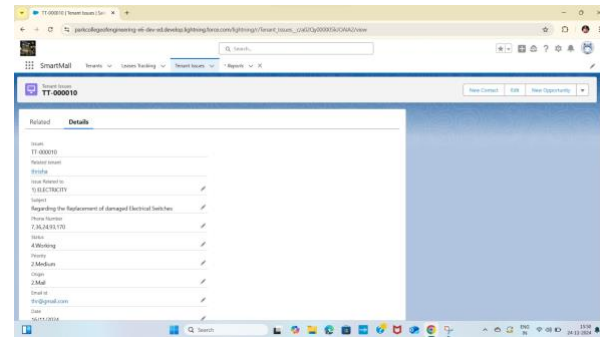


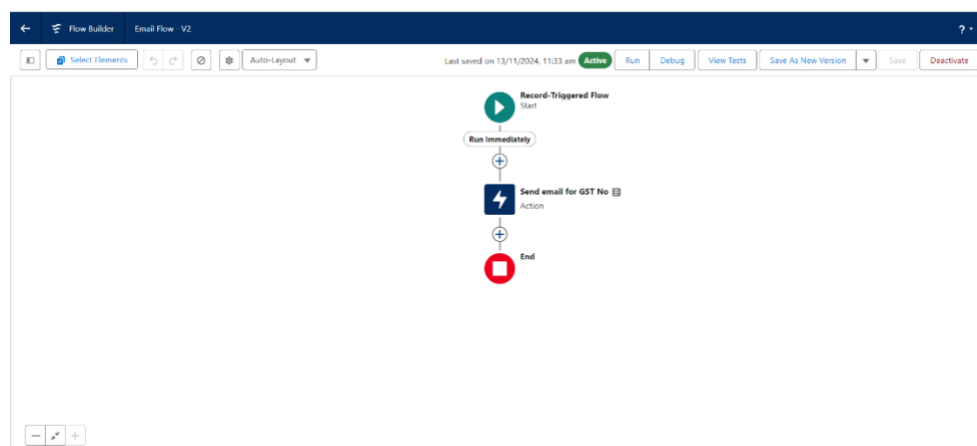
Figure:14



9. Create Flows

- 1) To create a flow click on setup==> Flow ==> Click on New Flow==> Select Record Triggered Flow
- 2) In Trigger the flow when Select A record is created
- 3) Select Condition required All Conditions are met (AND)
- 4) Select Field - GST_No__c , Operator - Is Null , Value - True.
- 5)Optimize the Flow for: Actions and Related Records
- 6)Add Element and choose ACTION in the search bar Search Send Email.

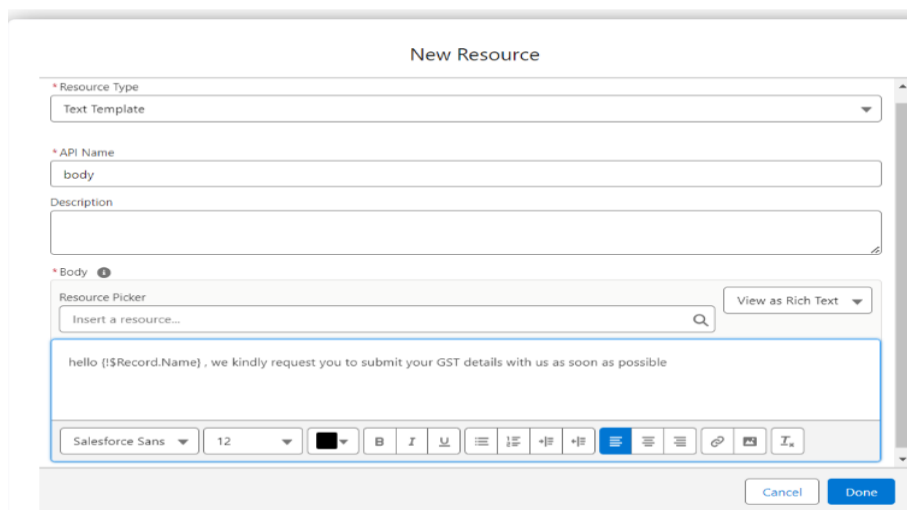
Figure:15



7) Label Name - Send email for Gst no, Description - This email is to alert the tenant that he or she has not submitted the GST NO yet.

8) Include Body And Create a Resource - Text Template As below-

Figure:16



New Resource

* Resource Type
Text Template

* API Name
body

Description

* Body ⓘ
Resource Picker
Insert a resource... View as Rich Text

hello {!\$Record.Name} , we kindly request you to submit your GST details with us as soon as possible

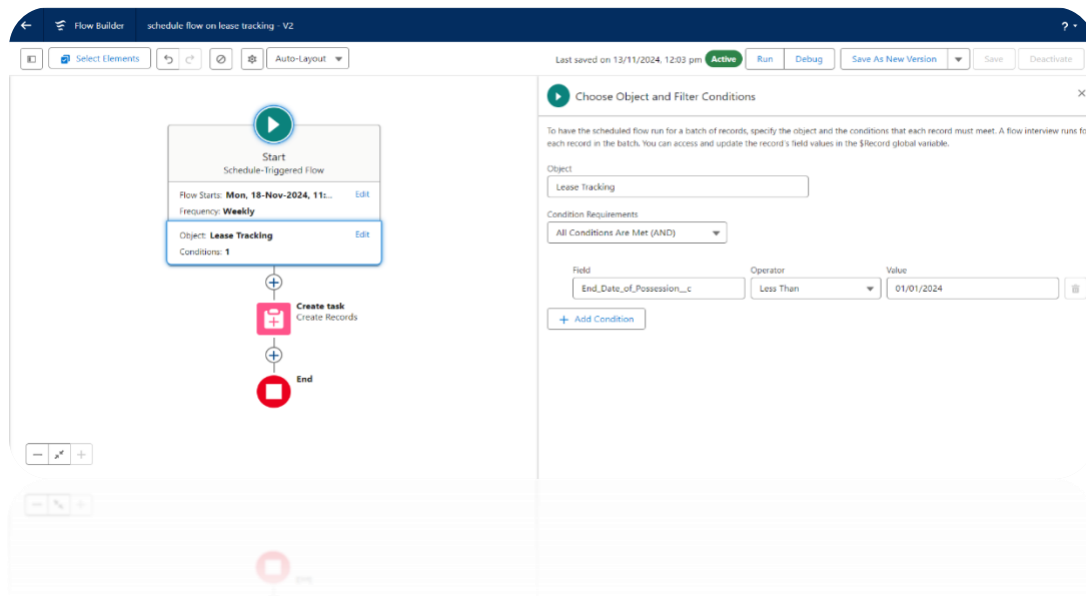
Salesforce Sans 12 B I U

Cancel Done

- **Record-Triggered Flow on Tenant Object:**

Automatically update the tenant's status based on certain conditions, such as lease renewal or overdue payments.

Figure:17



- **Scheduled Flow on Lease Management Object:** Sends reminders for upcoming lease expirations to tenants and administrators.

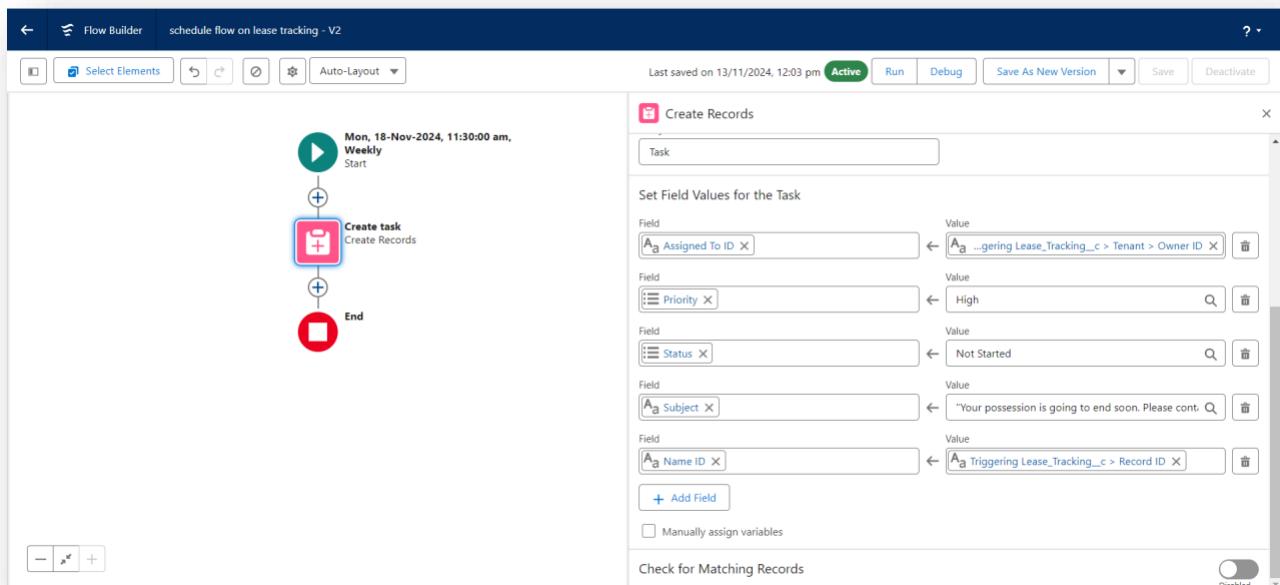
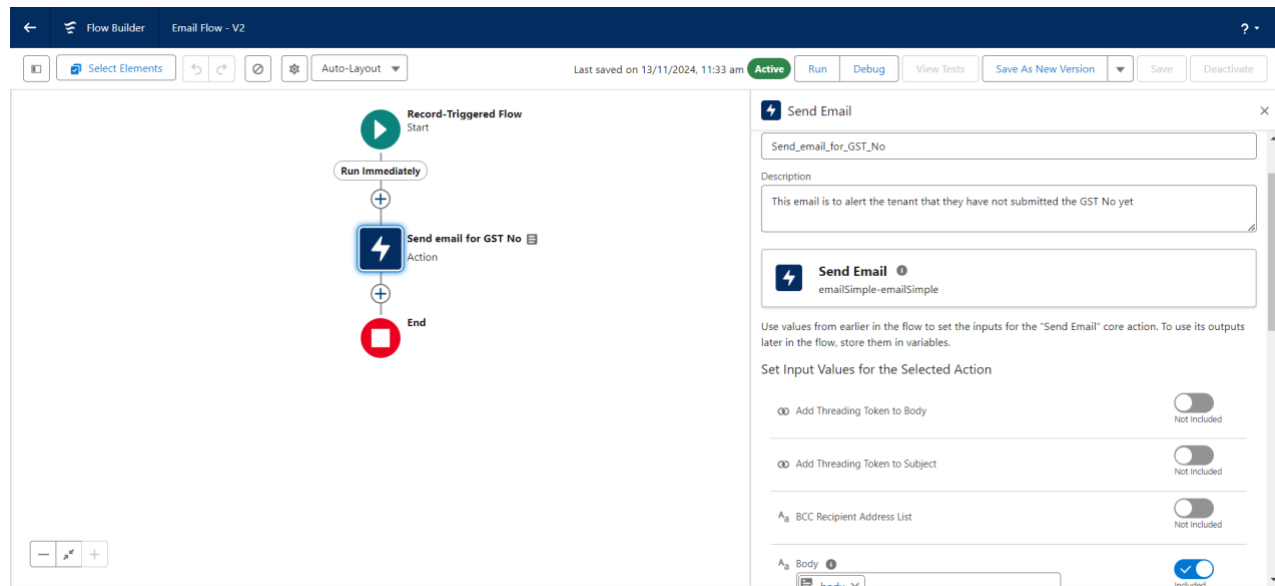


Figure:18

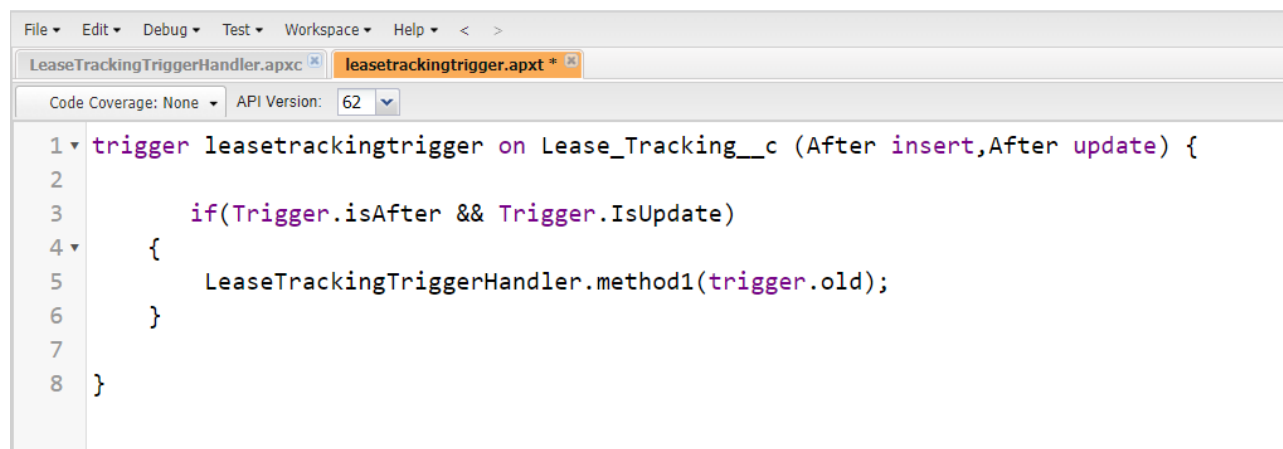
Figure:19



10. Apex Triggers

- **Trigger to Send Email for Unpaid Rent:** Automatically notify tenants if 50% of the rent remains unpaid after a specific period.

Figure :20



```

1 trigger leasetrackingtrigger on Lease_Tracking__c (After insert,After update) {
2
3     if(Trigger.isAfter && Trigger.IsUpdate)
4     {
5         LeaseTrackingTriggerHandler.method1(trigger.old);
6     }
7
8 }
  
```

- 1) Click on the gear icon and click on the developer console.
- 2) Click on file select New Apex Trigger
- 3) Name- leasetrackingtrigger, Object —> Lease_Tracking__c
- 4) Use Event - After insert and After Update and Use Trigger Context Variables as IsAfter and IsUpdate.

Trigger : -

CODE SNIPPET :

```
trigger leasetrackingtrigger on Lease_Tracking__c (After insert,After update) {  
    if(Trigger.isAfter && Trigger.IsUpdate)  
    {  
        LeaseTrackingTriggerHandler.method1(trigger.old);  
    }  
  
}
```

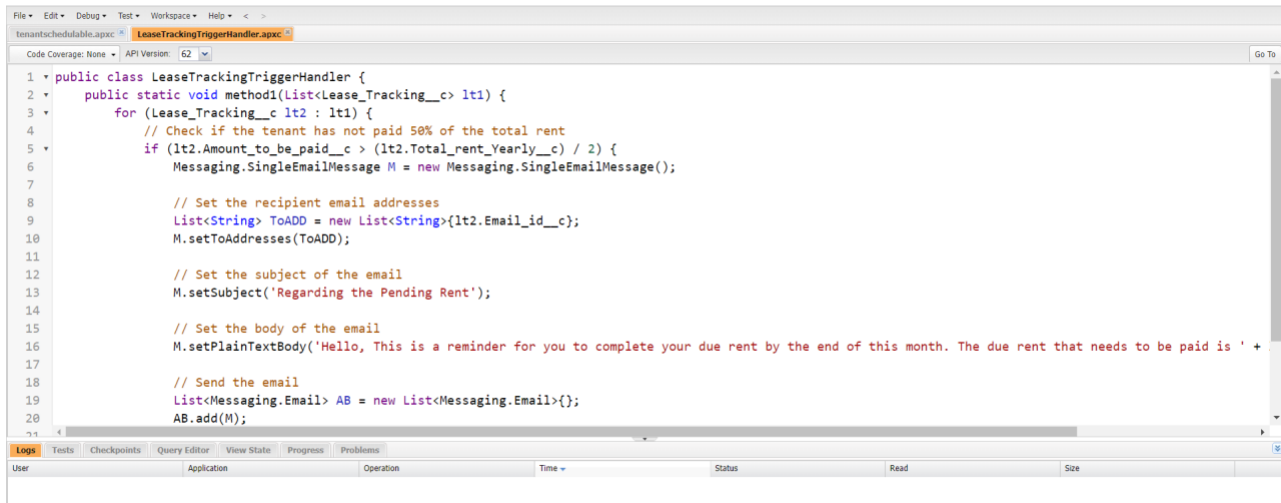
Trigger Handler: -

1) Create an apex class and Name it LeaseTrackingTriggerHandler

CODE SNIPPET : -

```
public class LeaseTrackingTriggerHandler {  
  
    public static void method1(List<Lease_Tracking__c> lt1)  
    {  
        for(Lease_Tracking__c lt2: lt1 )  
        {  
            if(lt2.Amount_to_be_paid__c > (lt2.Total_rent_Yearly__c)/2)  
            {  
                Messaging.SingleEmailMessage M = New Messaging.SingleEmailMessage();  
  
                List<String> ToADD = New List<String>{lt2.Email_id__c};  
  
                M.setToAddresses(ToADD);  
                M.setSubject('Regarding the Pending Rent');  
                M.setPlainTextBody('Hello, This is an Reminder for you to complete your due  
rent by the end this month, your due rent thatneeds to be paid is' +lt2.Amount_to_be_paid__c);  
  
                List<Messaging.Email> AB = New List<Messaging.Email>{ };  
                AB.add(M);  
                Messaging.sendEmail(AB);  
            }  
        }  
    }  
}
```


Figure :21



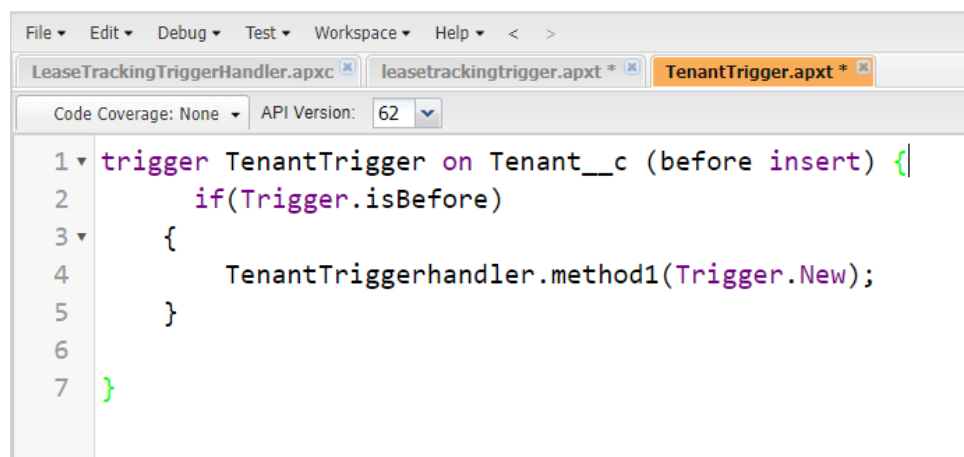
```

1 public class LeaseTrackingTriggerHandler {
2     public static void method1(List<Lease_Tracking__c> lt1) {
3         for (Lease_Tracking__c lt2 : lt1) {
4             // Check if the tenant has not paid 50% of the total rent
5             if (lt2.Amount_to_be_paid__c > (lt2.Total_rent_Yearly__c) / 2) {
6                 Messaging.SingleEmailMessage M = new Messaging.SingleEmailMessage();
7
8                 // Set the recipient email addresses
9                 List<String> ToADD = new List<String>{lt2.Email_id__c};
10                M.setToAddresses(ToADD);
11
12                // Set the subject of the email
13                M.setSubject('Regarding the Pending Rent');
14
15                // Set the body of the email
16                M.setPlainTextBody('Hello, This is a reminder for you to complete your due rent by the end of this month. The due rent that needs to be paid is ' +
17
18                // Send the email
19                List<Messaging.Email> AB = new List<Messaging.Email>{};
20                AB.add(M);

```

- **Trigger to Validate PAN Card:** Ensures the PAN number entered is valid, throwing an error otherwise.
 - 1) Click on the gear icon and click on the developer console.
 - 2) Click on file select New Apex Trigger
 - 3) Name- TenantTrigger, Object - Tenant
 - 4) Use Events - Before insert and Trigger context Variable – IsBefore

Figure:22



```

1 trigger TenantTrigger on Tenant__c (before insert) {
2     if(Trigger.isBefore)
3     {
4         TenantTriggerhandler.method1(Trigger.New);
5     }
6
7 }

```

Trigger : -

CODE SNIPPET :

```
trigger TenantTrigger on Tenant__c (before insert) {  
  
    if(trigger.isBefore)  
    {  
        TenantTriggerhandler.method1(trigger.New);  
    }  
  
}
```

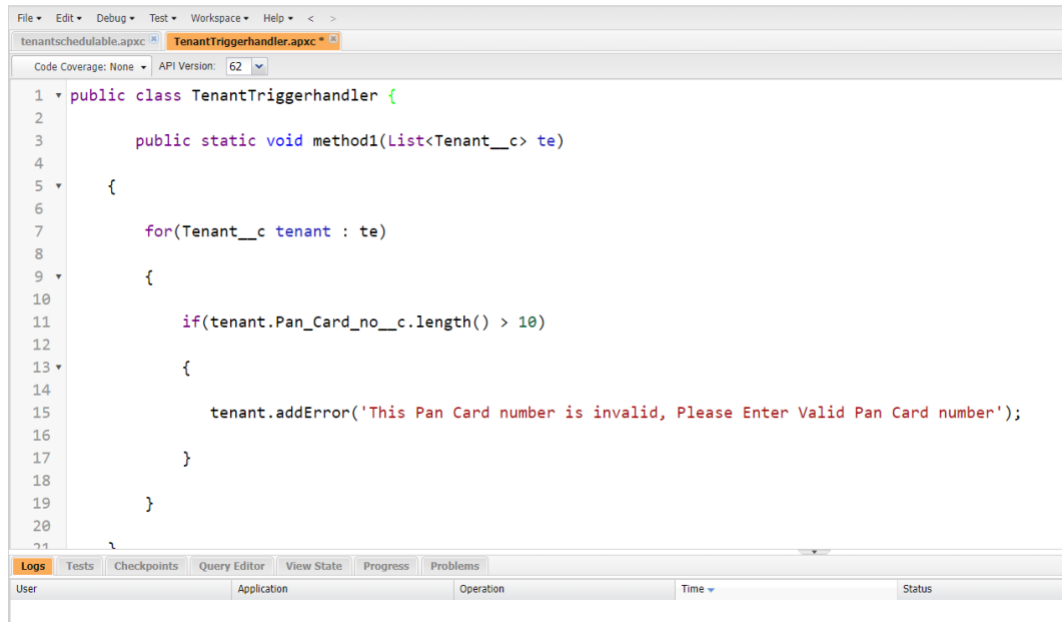
Trigger Handler : -

1) Create an apex class and Name it TenantTriggerhandler

CODE SNIPPET: -

```
public class TenantTriggerhandler {  
  
    public static void method1(List<Tenant__c> te)  
  
    {  
  
        for(Tenant__c tenant : te)  
  
        {  
  
            if(tenant.Pan_Card_no__c.length() > 10)  
  
            {  
  
                tenant.addError('This Pan Card number is invalid, Please Enter Valid Pan Card number');  
  
            }  
  
        }  
  
    }  
  
}
```

Figure :23



11. Asynchronous Apex

Schedule Apex: Batch process tenant payment data to generate a daily summary for the management team.

Delete the Tenant Records Monthly whose Status Of Possession is closed.

- 1) Create a class with name tenantschedulable
- 2) Give extension Schedulable to the class.
- 3) Open the Anonymous Window.
- 4) Schedule the class-

```

tenantschedulable a = new tenantschedulable();
string cron = '0 0 0 1 * ? *';
system.schedule('Delete the records monthly', cron, a);

```

CODE SNIPPET :-

```

public class tenantschedulable implements Schedulable
{
    public void execute(Schedulablecontext sc)
    {
        list<Tenant__c> ten = [SELECT Id, Status_of_Possession__c FROM Tenant__c ];

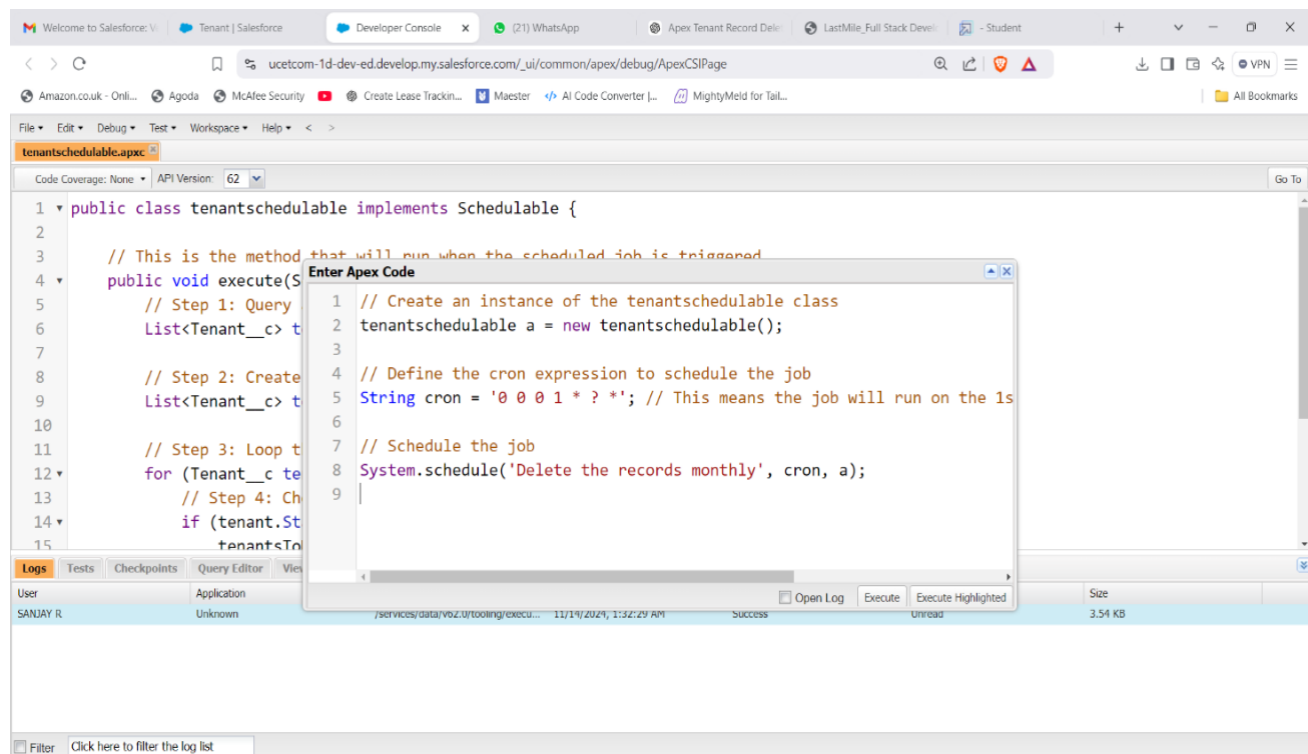
```

```
list<Tenant__c> tenantstodelete = New List<Tenant__c>();

for(Tenant__c te: ten)
{
    if(te.Status_of_Possession__c == 'Closed')
    {
        tenantstodelete.add(te);
    }
}

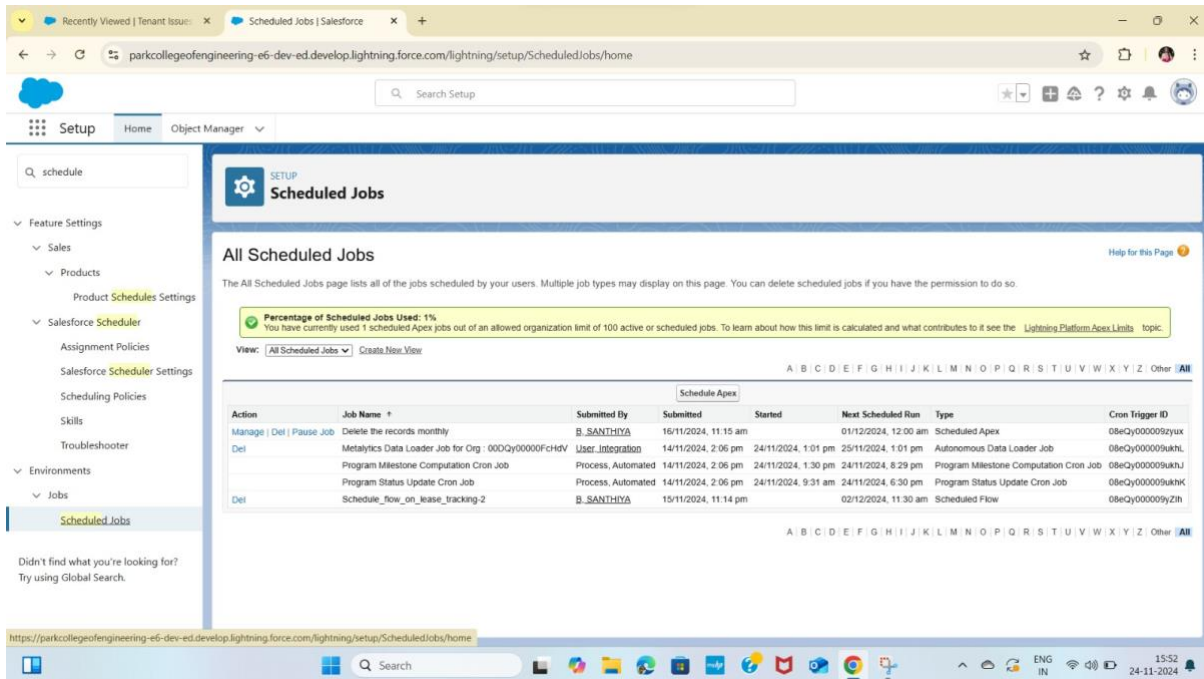
Delete tenantstodelete;
}
}
```

Figure :24



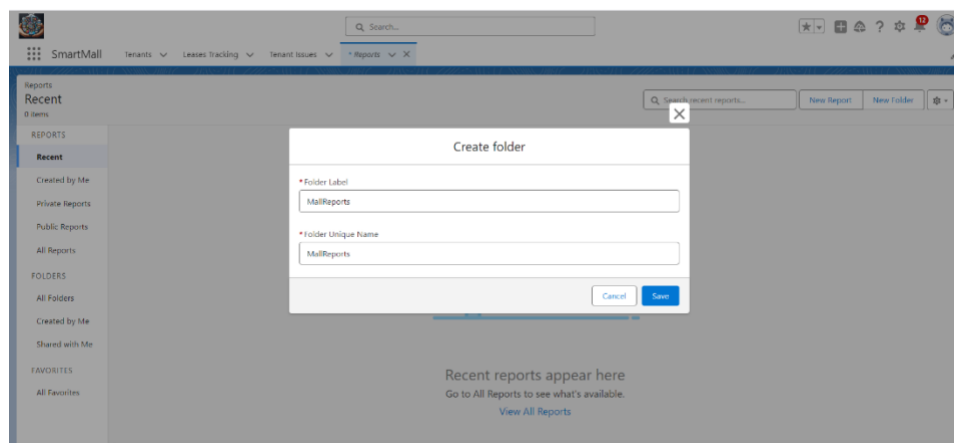
All Jobs are scheduled at correctly at 12:00 AM

Figure :25



12. Create Reports and Dashboards

Figure:26



- **Lease Management Records Report:** Displays lease details, including active and expired leases.

Figure:27

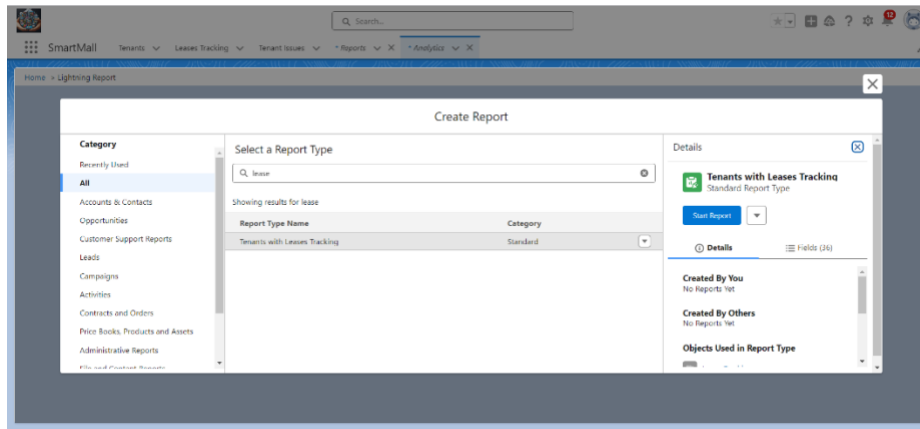


Figure:28

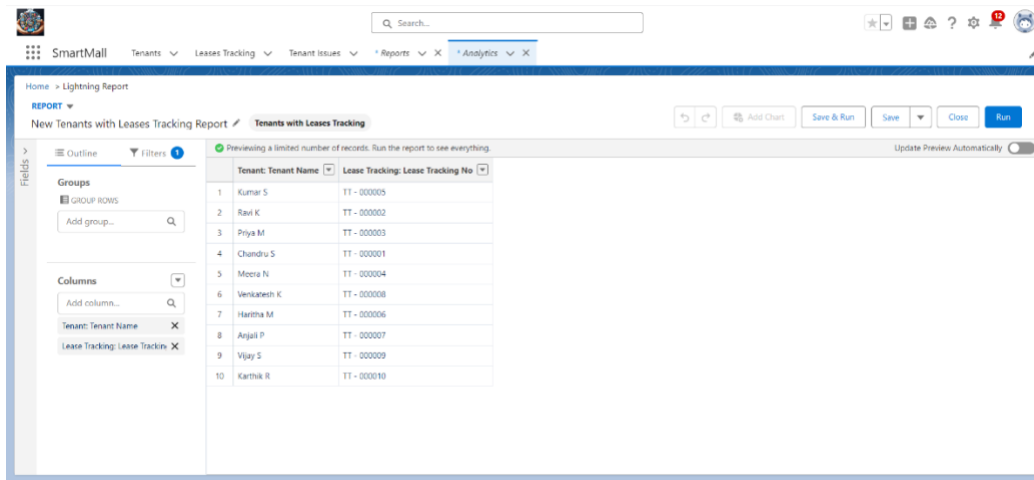


Figure:29

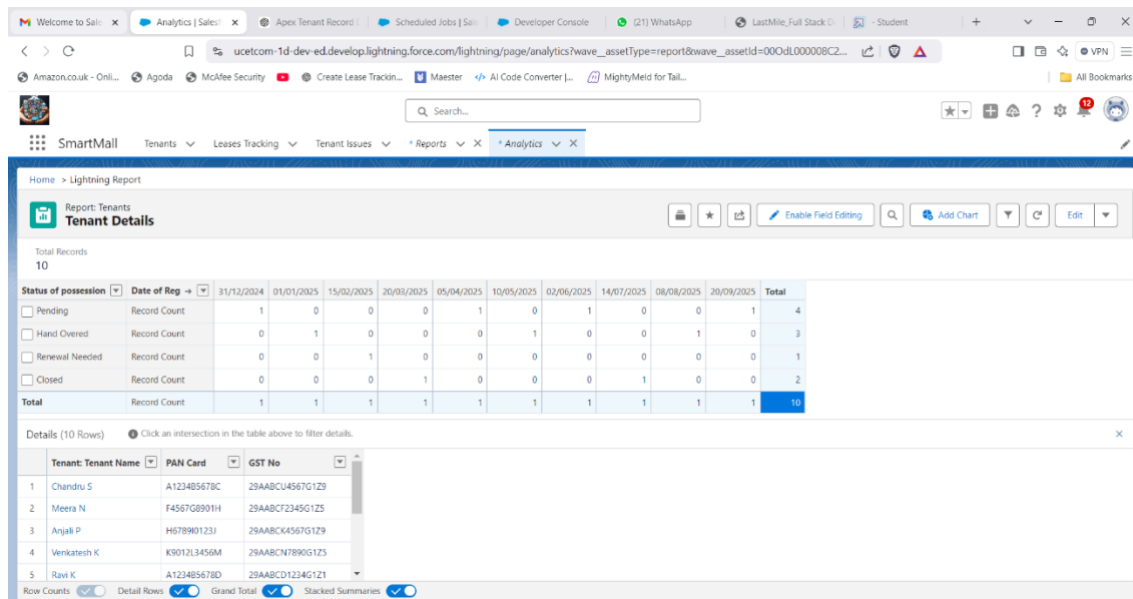
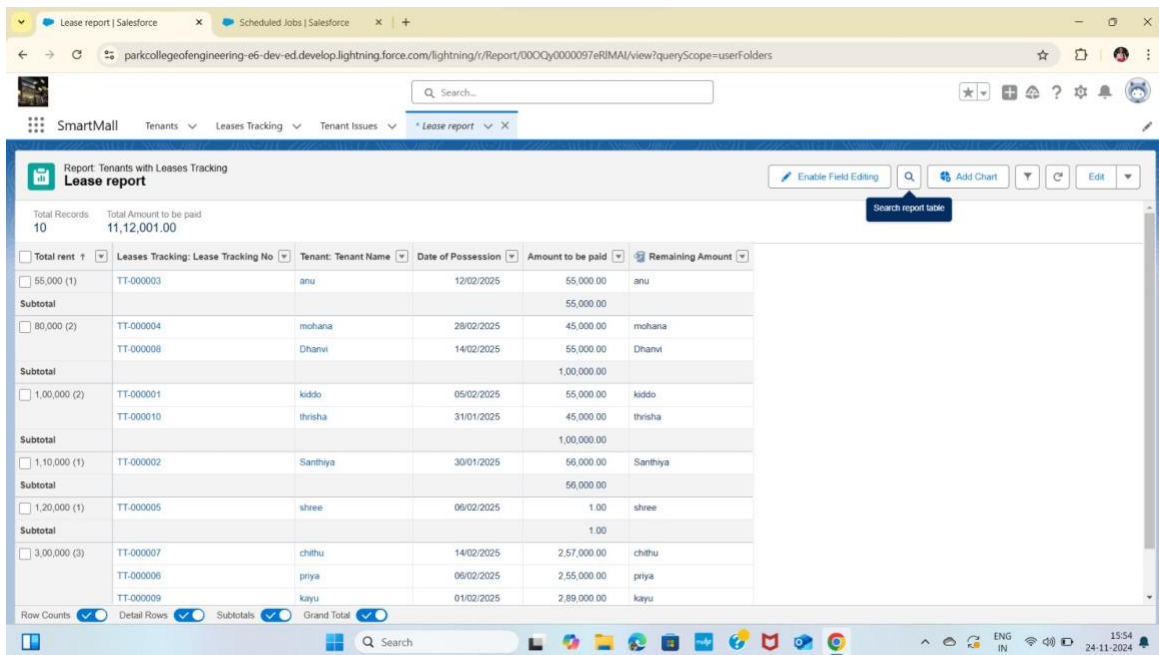


Figure:30



Report: Tenants with Leases Tracking
Lease report

Total Records: 10 | Total Amount to be paid: 11,12,001.00

Total rent	Leases Tracking: Lease Tracking No	Tenant: Tenant Name	Date of Possession	Amount to be paid	Remaining Amount
55,000 (1)	TT-000003	anu	12/02/2025	55,000.00	anu
Subtotal				55,000.00	
80,000 (2)	TT-000004	mohana	28/02/2025	45,000.00	mohana
	TT-000008	Dhanvi	14/02/2025	55,000.00	Dhanvi
Subtotal				1,00,000.00	
1,00,000 (2)	TT-000001	kiddo	05/02/2025	55,000.00	kiddo
	TT-000010	thrisha	31/01/2025	45,000.00	thrisha
Subtotal				1,00,000.00	
1,10,000 (1)	TT-000002	Santhiya	30/01/2025	56,000.00	Santhiya
Subtotal				56,000.00	
1,20,000 (1)	TT-000005	shree	05/02/2025	1.00	shree
Subtotal				1.00	
3,00,000 (3)	TT-000007	chithu	14/02/2025	2,57,000.00	chithu
	TT-000006	priya	06/02/2025	2,55,000.00	priya
	TT-000009	kayyu	01/02/2025	2,89,000.00	kayyu
Grand Total					

- **Tenant Issue Records Report:** Summarizes tenant complaints and their resolution statuses.
- **Custom Dashboards:** Visualize data for occupancy rates, tenant satisfaction, and lease renewal schedules.

Figure:31

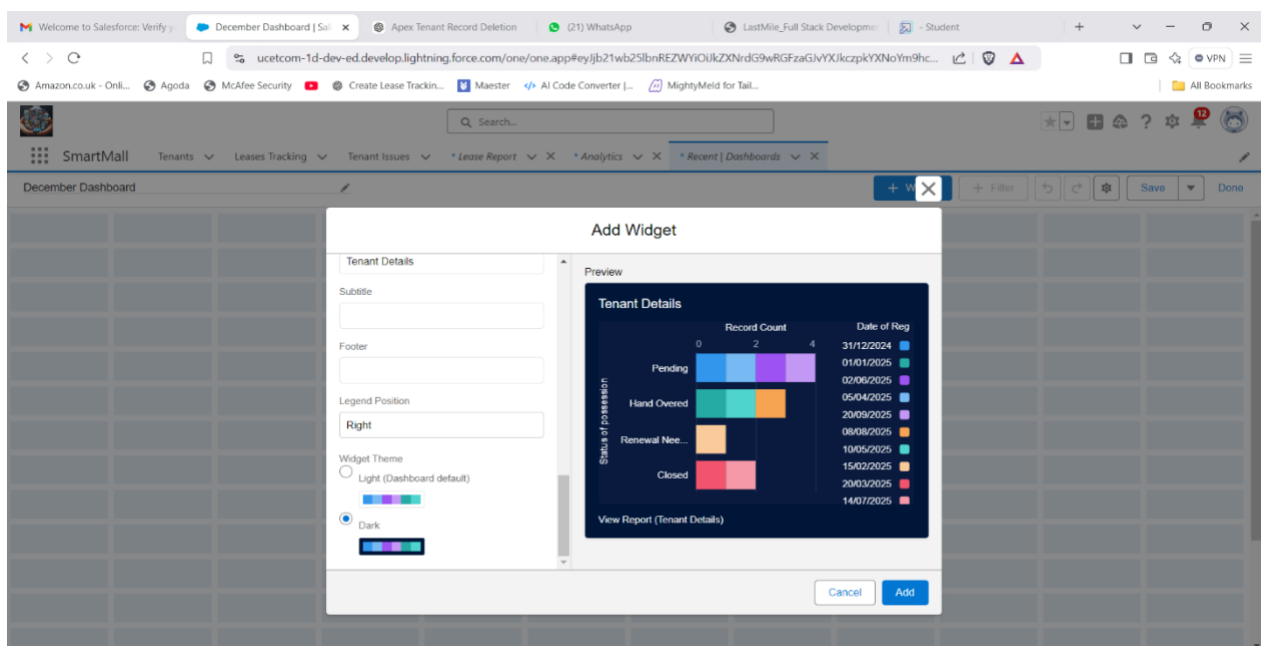


Figure:32

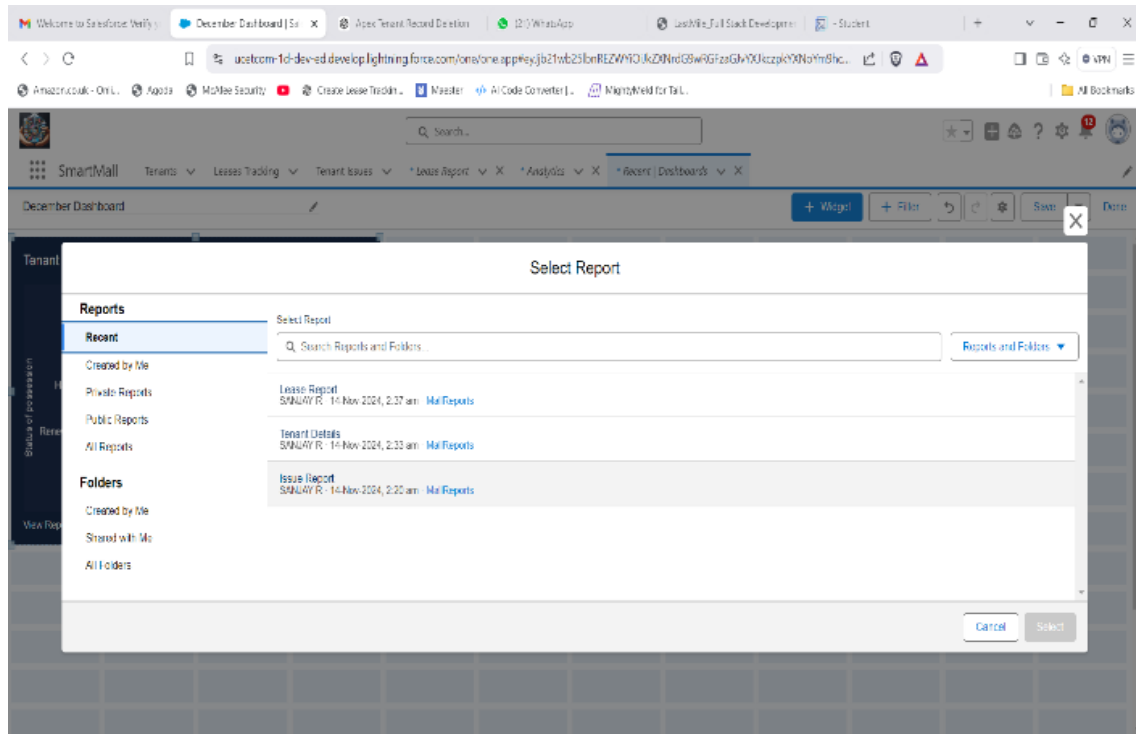
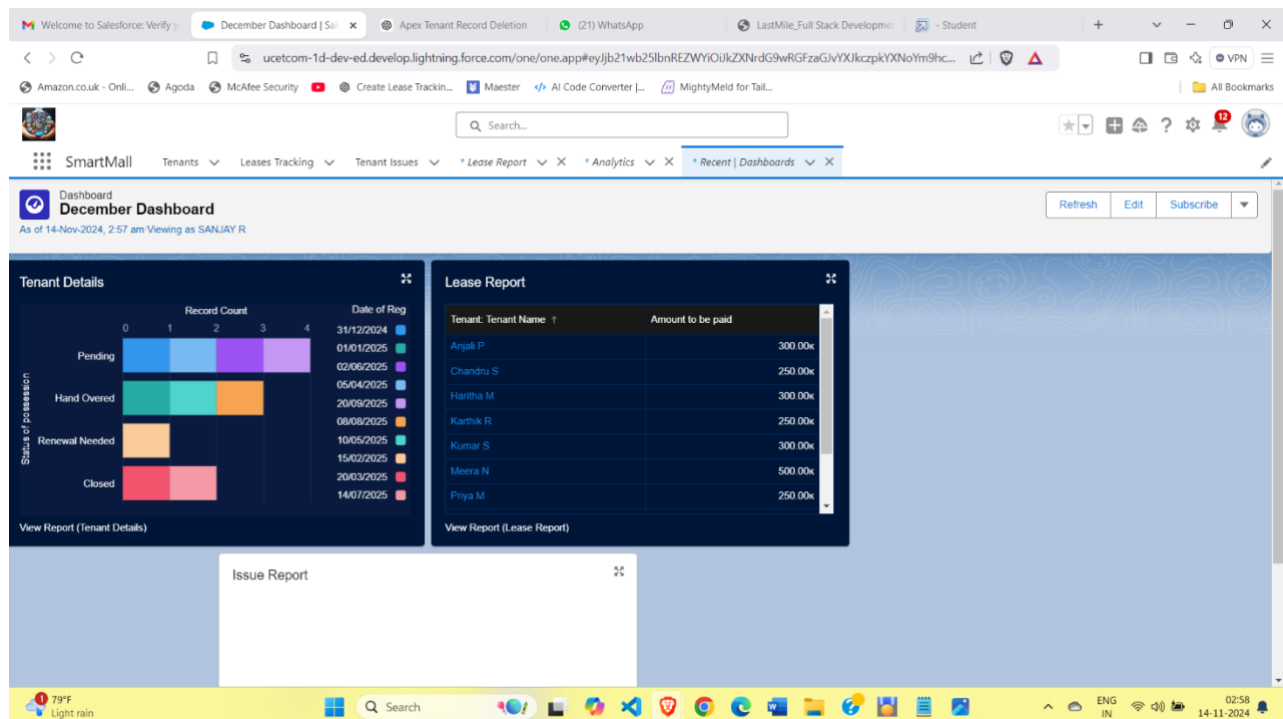


Figure:33



5. Testing and Validation

The application underwent multiple rounds of testing to ensure functionality:

- **Unit Testing:**
 - Validated Apex triggers , Apex classes , workflows, and data relationships.
- **UI Testing:**
 - Ensured seamless navigation and data accuracy in Lightning pages.
- **Integration Testing:**
 - Confirmed smooth operation between custom objects and workflows.
- **Stress Testing:**
 - Evaluated system performance under high data loads to ensure smooth operation with hundreds of concurrent records.
- **End-User Testing:**
 - Conducted testing sessions with potential users (mall administrators) to gather feedback on usability and interface design.
- **Regression Testing:**
 - Verified that newly added functionalities, such as record-triggered flows, did not affect existing workflows.

By addressing edge cases and refining the application based on user feedback, the app ensures a seamless and efficient experience for its users

6. Key Scenarios Addressed by Salesforce in the Implementation Project

- **Lease Renewal Notifications:** Alerts for leases nearing expiration.
- **Tenant Issue Resolution:** Centralized system for complaint tracking and resolution.
- **Financial Overview:** Dashboards showing rent collection progress and overdue accounts.
- **Rent Collection Alerts:** Automates reminders for overdue rent payments, ensuring timely revenue collection and reducing manual follow-ups.
- **Data Visualization:** Interactive dashboards provide insights into tenant retention rates, complaint resolution times, and lease expiration trends.
- **Tenant Onboarding:** Simplifies the process of onboarding new tenants with pre-configured templates for lease agreements and contact records.
- **Escalation Handling:** Implements workflows to escalate unresolved tenant complaints to higher management, ensuring accountability and swift resolution.

These scenarios ensure that all aspects of mall management are addressed, from tenant interactions to financial oversight.

7. Conclusion

The Management App successfully integrates Salesforce's powerful CRM capabilities with the unique requirements of mall management. By automating lease tracking, tenant communication, and reporting processes, the application ensures enhanced operational efficiency and improved tenant satisfaction. This project demonstrates the adaptability of Salesforce for building comprehensive industry-specific CRM solutions.