COLLEGE CODE: 9520

P.S.R.R. COLLEGE OF ENGINEERING

SIVAKASI - 626140

VIRUDHUNAGAR DISTRICT

DEPARTMENT

OF

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Project – Lease Management

Team ID : NM2025TMID02660

Team Size : 4

Team Leader : Francies T

Email: franciest2004@gmail.com

Team member 1 : Gurumoorthi B

Email: gurumoorthy0005@gmail.com

Team member 2 : Santhana Kumar R

Email: Kumar1122004@gmail.com

Team member 3 : BAL Abisheak C

Email: skyshk555@gmail.com

PROJECT TITLE: **LEASE MANAGEMENT**

Phase 1: Ideation Phase

Project Title: Lease Management System using Salesforce

Introduction:

The **Lease Management System** is a Salesforce-based application designed to efficiently handle and automate leasing operations for properties. It simplifies the process of managing property details, tenants, lease agreements, and payment transactions — all within a centralized and secure cloud environment.

Salesforce provides a robust platform with powerful automation, workflow management, and reporting capabilities that make it ideal for developing business process–driven applications like lease management.
The project aims to digitize and streamline manual leasing operations, ensuring accuracy, transparency, and timely communication between tenants and property owners.

Problem Statement:

Traditional lease management is often handled manually using spreadsheets, paper documents, or basic systems, leading to:

- Difficulty tracking lease start and end dates.
- Errors in payment and tenant data.
- Poor communication between tenants and property managers.
- Lack of automated reminders for rent payments or lease expirations.
- Inefficient record management and approval processes.

Hence, there is a need for a digital system that automates these repetitive processes, reduces human error, and ensures data security.

Objectives:

The main objectives of the Lease Management System are:

- To automate lease, tenant, and payment record management using Salesforce.
- To implement approval workflows for tenant leave or property vacancy requests.
- To send automated emails for lease approvals, payment reminders, and confirmations.
- To maintain centralized, real-time access to all lease-related information.

- To enhance operational efficiency and ensure transparent communication.

Scope of the Project:

This project focuses on managing the complete lease lifecycle, including:

- Property Management: Adding and maintaining property details.
- Tenant Management: Recording tenant profiles, contact information, and lease status.
- Lease Tracking: Monitoring lease start/end dates with validation rules.
- Payment Management: Recording payments, verifying status, and automating monthly reminders.
- Approval Workflows: Automating leave approval/rejection for tenants.
- Email Automation: Sending personalized notifications using Salesforce Email Templates and Flows.

The system is built entirely within Salesforce's declarative tools (objects, fields, automation, and approvals) with limited Apex code for validation and scheduling.

Tools and Technologies Used:

Category	**Technology**
Platform	Salesforce (Developer Edition)
Language	Apex (for triggers and schedulers)
Automation Tools	Salesforce Flow, Approval Process
Communication Tools	Email Templates, Email Alerts
Database	Salesforce Object Database
UI Components	Lightning App Builder, Tabs, Page Layouts

Expected Outcome:

By the end of this project, the following outcomes are expected:

- A fully functional Salesforce Lease Management Application capable of handling tenants, properties, leases, and payments.
- Automated email reminders for monthly payments and lease expirations.
- Implemented approval processes for tenant leave requests.

- Error-free data management with validation and lookup relationships.
- Improved efficiency and user satisfaction through workflow automation.

Benefits:

- Reduced manual work and data redundancy.
- Increased transparency between property managers and tenants.
- Timely reminders and updates via automated emails.
- Secure cloud storage and easy accessibility of lease records.
- Real-time dashboards and reports for better decision-making.

Phase 2: Project Planning Phase

Objective:

The main objective of this phase is to plan and design the overall structure, functionality, and execution flow of the Lease Management System in Salesforce. The planning phase ensures that all technical, functional, and business requirements are clearly defined and aligned with the project goals.

Project Scope:

The **Lease Management System** is designed to streamline the process of managing rental properties, tenants, lease agreements, and payments. It provides an automated workflow that handles lease approval, payment reminders, and tenant notifications using Salesforce automation tools like **Flows, Approval Processes, Apex Triggers, and Scheduled Jobs**.

In Scope:

- Managing property, tenant, lease, and payment records
- Automated email communication for payments and lease approvals
- Validation and approval processes for lease requests
- Payment tracking and monthly reminders
- Restriction for multiple tenants under one property

Out of Scope:

- Integration with external payment gateways
- Mobile app integration

Tools and Technologies:

Tool / Technology	**Purpose**
Salesforce Developer Org	Cloud platform for CRM and app development
Apex Programming	Custom business logic implementation
Salesforce Flow	Automation for record-triggered actions
Approval Process	Automating lease and tenant approval
Email Templates	Sending automatic email notifications

Tool / Technology	Purpose
Lightning App Builder	UI and navigation setup
Object Manager	Creating and managing data model
Validation Rules	Ensuring data accuracy and consistency

Modules Identified:

Module Name	Description
Property Management	Handles property details including type, address, and size.
Tenant Management	Maintains tenant contact info, status, and assigned property.
Lease Management	Records lease start and end dates, and connects tenants with properties.
Payment Management	Tracks rent payments, amount, and payment status.
Automation Module	Implements approval workflows, email templates, and scheduled payment reminders.

System Design Overview:

1. Objects and Relationships

- Property (Custom Object) → Master for Lease
- Tenant (Custom Object) → Master for Payment
- Lease (Custom Object) → Linked with Property (Lookup)
- Payment (Custom Object) → Linked with Tenant (Lookup)

2. Relationships

- Master-Detail: Tenant ↔ Property
- Lookup: Lease ↔ Property, Payment ↔ Tenant

3. Validation Rules

- Example: **End_Date__c > Start_Date__c**
  → Ensures the lease end date is always after the start date.

Workflow and Automation Plan:

Automation Type	Description	Tools Used
Email Alerts	Sends automated emails for payment, approval, and rejection.	Email Templates
Approval Process	Approves or rejects tenant leave requests.	Approval Process
Flow Automation	Sends confirmation email when payment is marked as "Paid".	Record-Triggered Flow
Apex Trigger	Prevents duplicate property assignment to tenants.	Apex & Handler Class
Scheduled Apex	Sends monthly payment reminders automatically.	Apex Scheduler

Deliverables of Phase 2:

- Finalized data model (objects and fields)
- Identified automation workflows
- Approval process design plan
- Apex class and trigger planning
- System architecture and navigation layout

Conclusion:
The project planning phase successfully outlines the entire structure, automation flow, and Salesforce components required for the Lease Management System. With this plan, the development can proceed efficiently to the design and implementation stages.

Phase 3: Project Design Phase

Introduction:

The **Execution Phase** is where the actual implementation of the Lease Management System begins. This phase involves setting up the Salesforce Developer environment, creating required custom objects, defining relationships, adding fields, validation rules, email templates, and building automation workflows such as approval processes, triggers, and scheduled jobs.

The main goal of this phase is to convert the planned system design into a fully functional Salesforce application capable of managing property leases, tenants, payments, and approval workflows.

Objectives:

- To implement the design created during the Planning Phase.
- To configure Salesforce objects, fields, and relationships.
- To create automation components (flows, approval processes, and triggers).
- To test functionalities and ensure the system meets business requirements.
- To deliver a working Lease Management application in Salesforce.

Activities Carried Out:

1. Salesforce Developer Environment Setup

- Created a Salesforce Developer Account using the developer portal.
- Verified and activated the account through the provided email link.
- Logged into Salesforce Setup for environment configuration.

2. Object Creation

Custom Objects were created to represent core entities in the system:

Object Name	Description	Key Fields
Property	Stores property-related data such as name, address, and type.	Property Name, Address, Type, Sqft
Tenant	Maintains tenant details.	Tenant Name, Email, Phone, Status

Object Name	Description	Key Fields
Lease	Tracks lease agreements between tenants and property owners.	Lease Name, Start Date, End Date
Payment for Tenant	Manages monthly rent and payment status.	Payment Name, Payment Date, Amount, Status

3. Tab Creation

Tabs were created for easy navigation and record management for each custom object:

- Property Tab
- Tenant Tab
- Lease Tab
- Payment for Tenant Tab

Each tab provides a user interface for creating, viewing, and managing records related to that object.

4. Lightning App Creation

A **Lightning App named "Lease Management"** was created, combining all four objects under a single app workspace for easy access.

Steps involved:

1. Created a Lightning App from **App Manager**.
2. Added navigation items — Property, Tenant, Lease, and Payment for Tenant.
3. Assigned the app to **System Administrator profile**.
4. Customized branding and app color.

5. Field Creation

Custom fields were added to capture detailed information.

Object	Field Name	Data Type	Description
Property	**Address**	**Long Text**	**Property address**
Property	**Type**	**Picklist**	**Type of property (1BHK, 2BHK, 3BHK)**
Tenant	**Email**	**Email**	**Tenant's email address**

Object	Field Name	Data Type	Description
Tenant	**Status**	**Picklist**	**Stay / Leaving**
Lease	**Start Date**	**Date**	**Lease start date**
Lease	**End Date**	**Date**	**Lease end date**
Payment	**Amount**	**Number**	**Rent amount**
Payment	**Check for Payment**	**Picklist**	**Paid / Not Paid**

6. Relationship Creation

To ensure relational integrity, lookup and master-detail relationships were created:

Relationship	Type	Description
Lease → Property	Lookup	Links lease to property
Payment → Tenant	Lookup	Links payment to tenant
Tenant → Property	Master-Detail	Links tenant to property

7.Validation Rule

A validation rule was created in the **Lease Object** to ensure data accuracy:

Rule Name: lease_end_date

Formula: End_date__c < Start_date__c

Error Message: "End date must be greater than Start date."

This prevents invalid lease durations.

8. Email Templates

Custom Email Templates were created for automated communication:

1. **Tenant Leaving –** Request for approval of tenant’s leave.
2. **Leave Approved –** Confirmation of leave approval.
3. **Leave Rejected –** Notification of rejection.
4. **Monthly Payment Reminder –** Reminder to pay rent.
5. **Successful Payment Confirmation –** Acknowledgment of payment receipt.

9.Approval Process

Created for **Tenant Object** named “Check for Vacant.”
The approval process automates tenant leave approvals and integrates email notifications for approval, rejection, and submission actions.

Actions configured:

- **Initial Submission:** Sends “Tenant Leaving” email.
- **Final Approval:** Sends “Leave Approved” email.
- **Final Rejection:** Sends “Leave Rejected” email.

10. Apex Trigger and Handler

An **Apex Trigger (test)** and **Handler Class (testHandler**) were created to prevent assigning multiple tenants to the same property.

Trigger Logic:

- Runs before insert.
- Checks if a property already has a tenant.
- Displays an error: *“A tenant can have only one property.”*

11. Flow Automation

A **Record-Triggered Flow** named *monthly_payment* was created to send an automatic email when the payment status changes to “Paid.”

12. Scheduled Apex Class:

A **Scheduled Apex Class** (MonthlyEmailScheduler) was developed to automatically send rent reminders to all tenants on the 1st of every month.

Tools and Technologies Used:

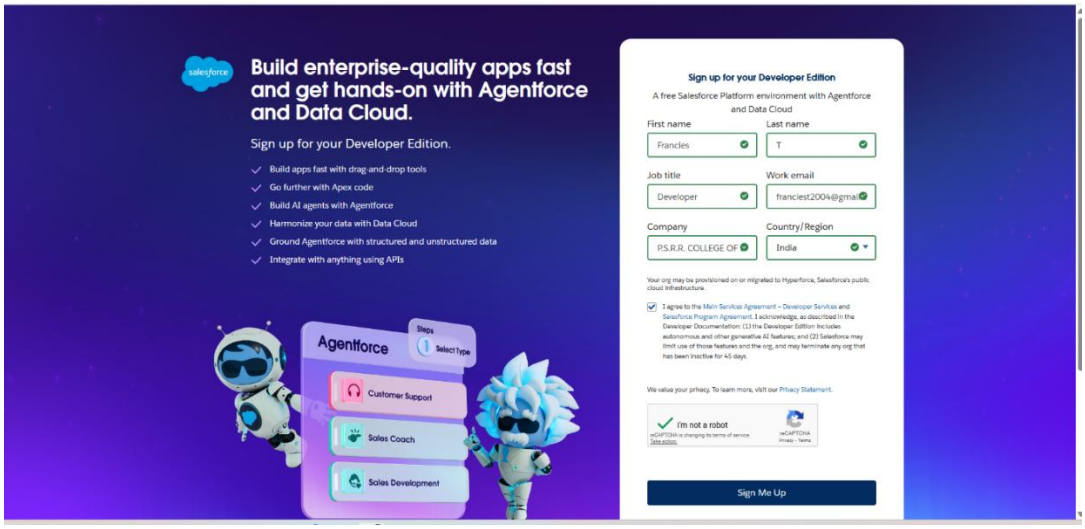
- Salesforce Platform (Lightning Experience)
- Apex Programming Language
- Salesforce Flow Builder
- Email Templates & Process Builder
- Cloud Database (Salesforce Objects)

Deliverables

- Fully functional Lease Management Salesforce Application.
- Configured Objects, Fields, Tabs, and Relationships.
- Working Automation Components (Flows, Approvals, Triggers).
- Email Notification System.
- Monthly Payment Reminder System via Scheduler.

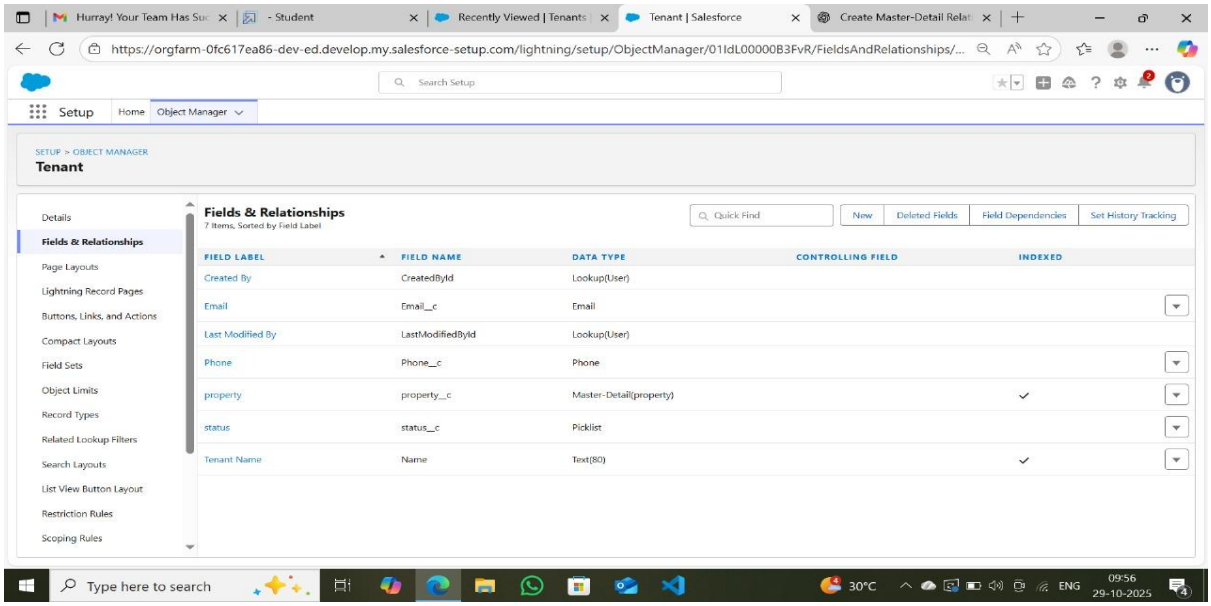
Output Screenshots:

1. Developer Org Setup:

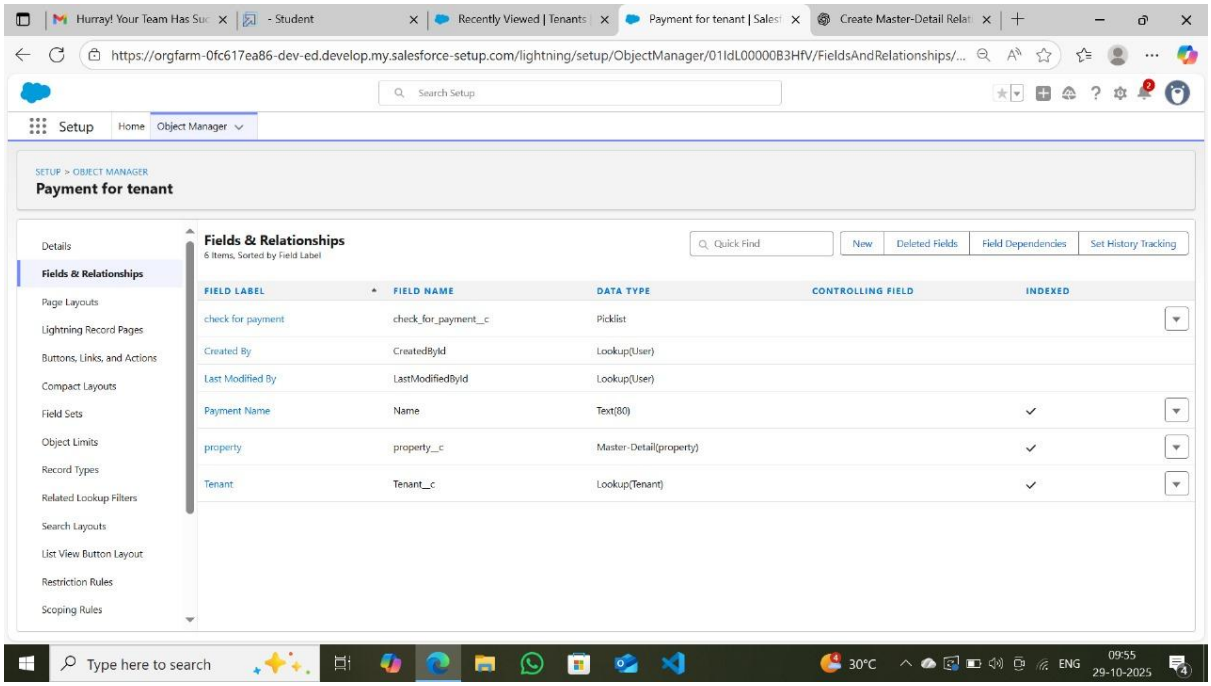


2. Object and Field Creation:

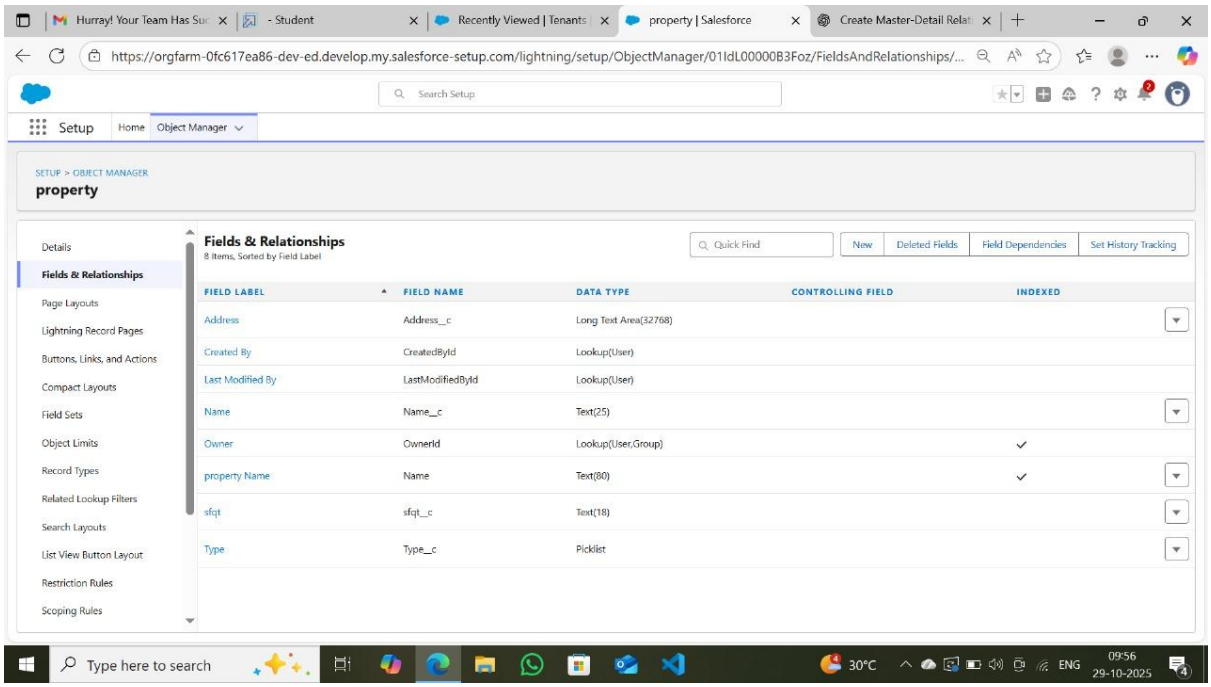
Tenant:



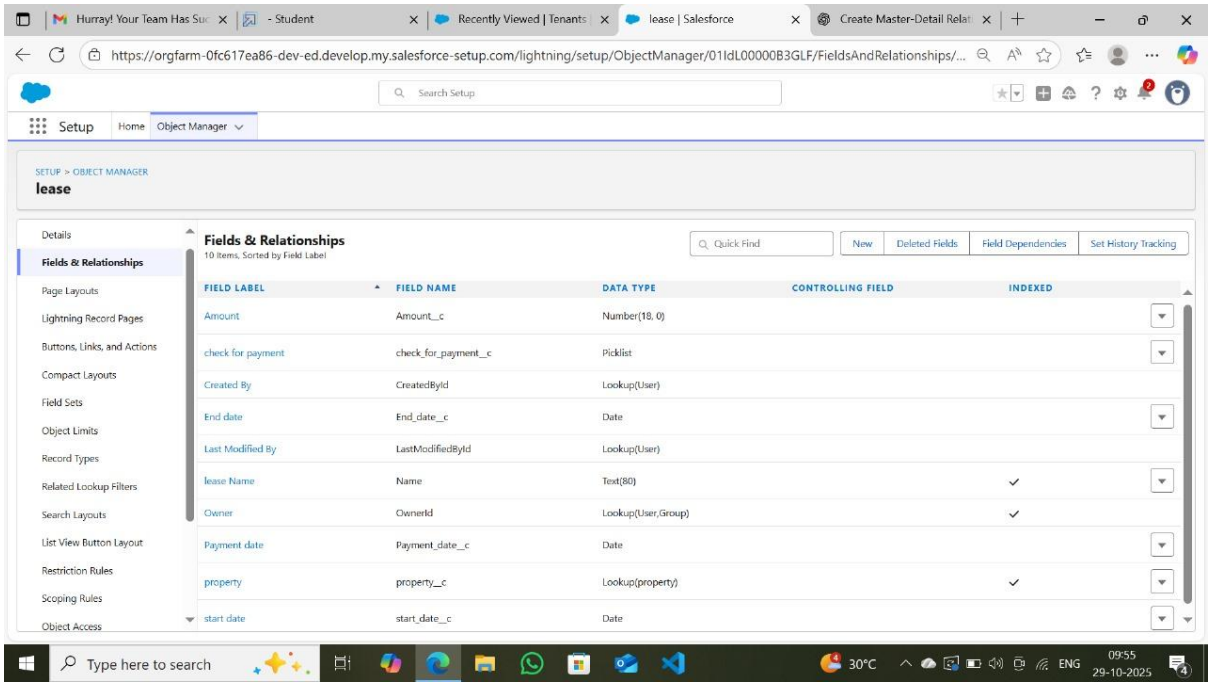
Payment for Tenant:



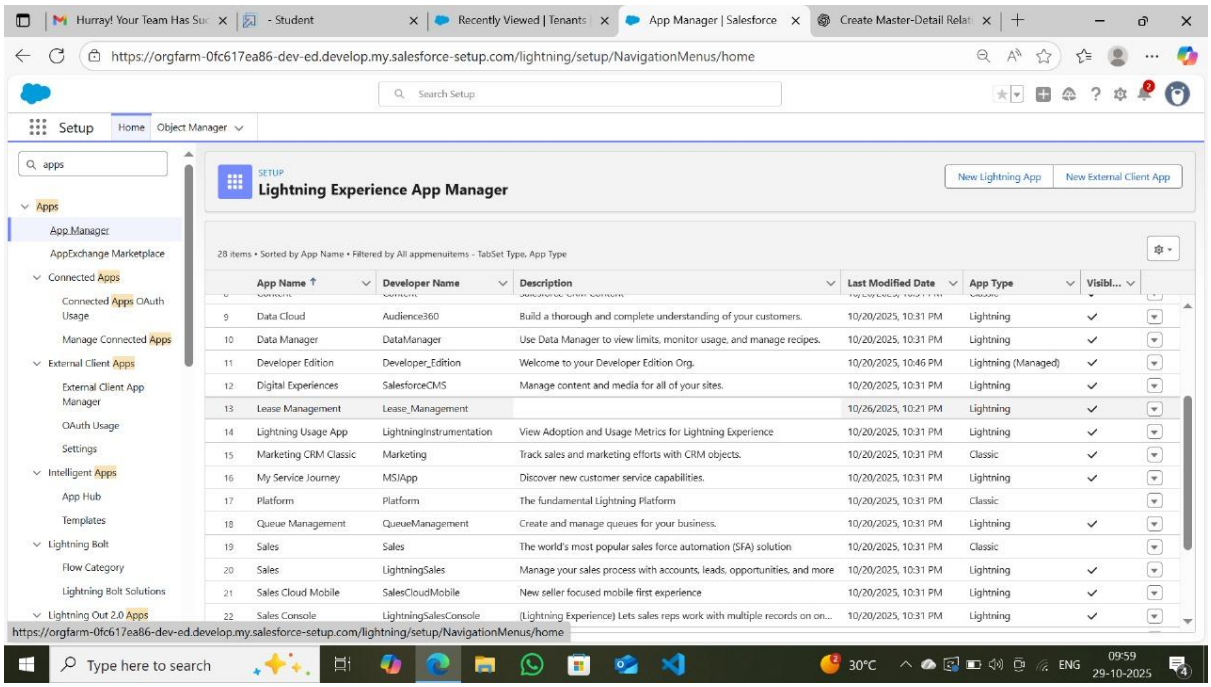
Property:



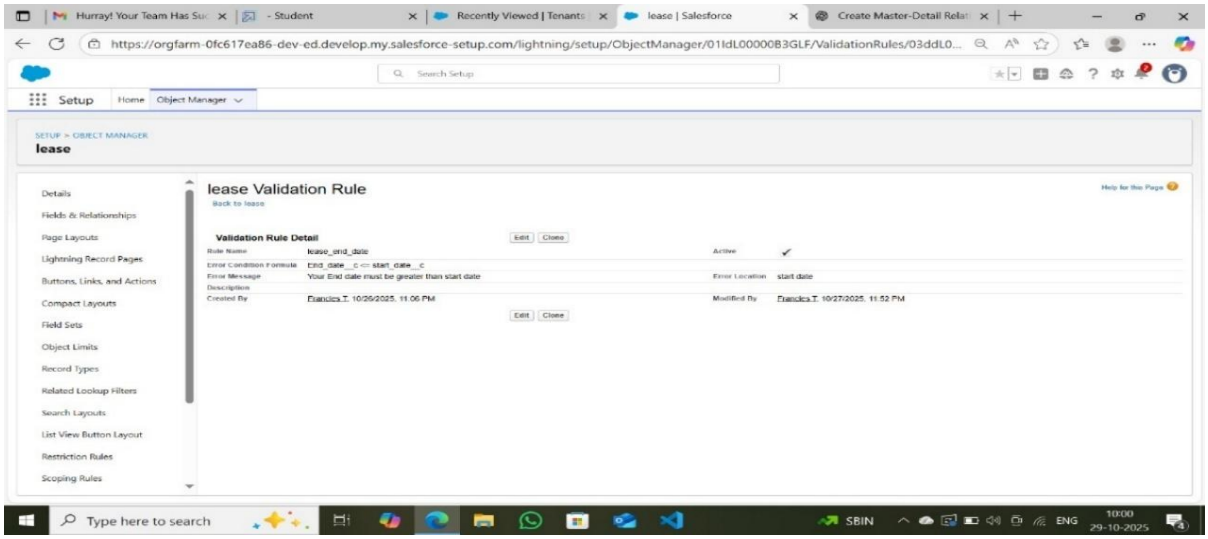
Lease:



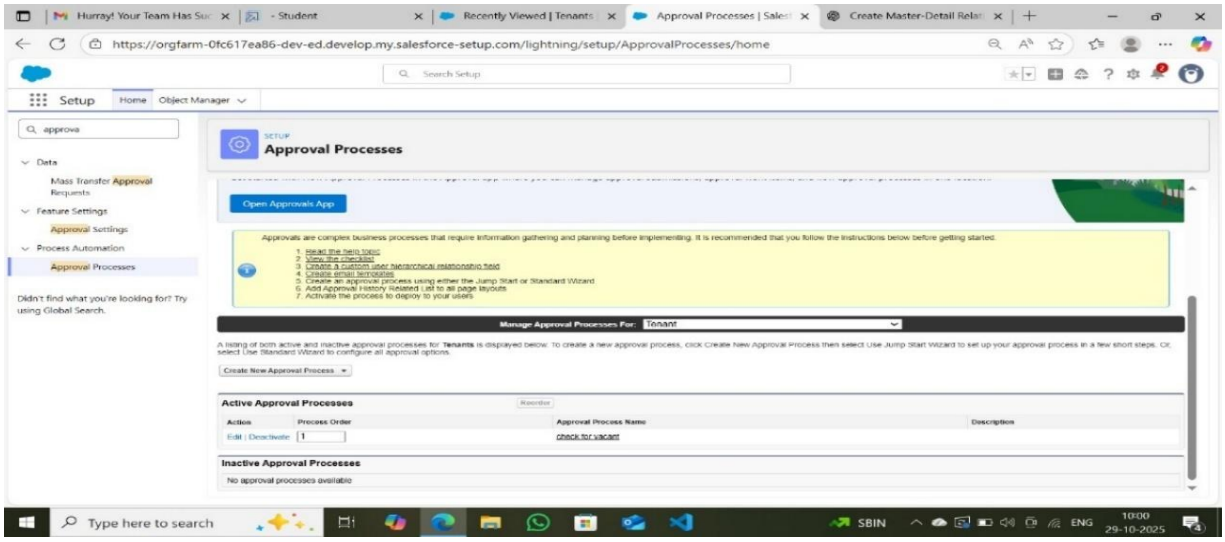
3.Lightning App Interface:



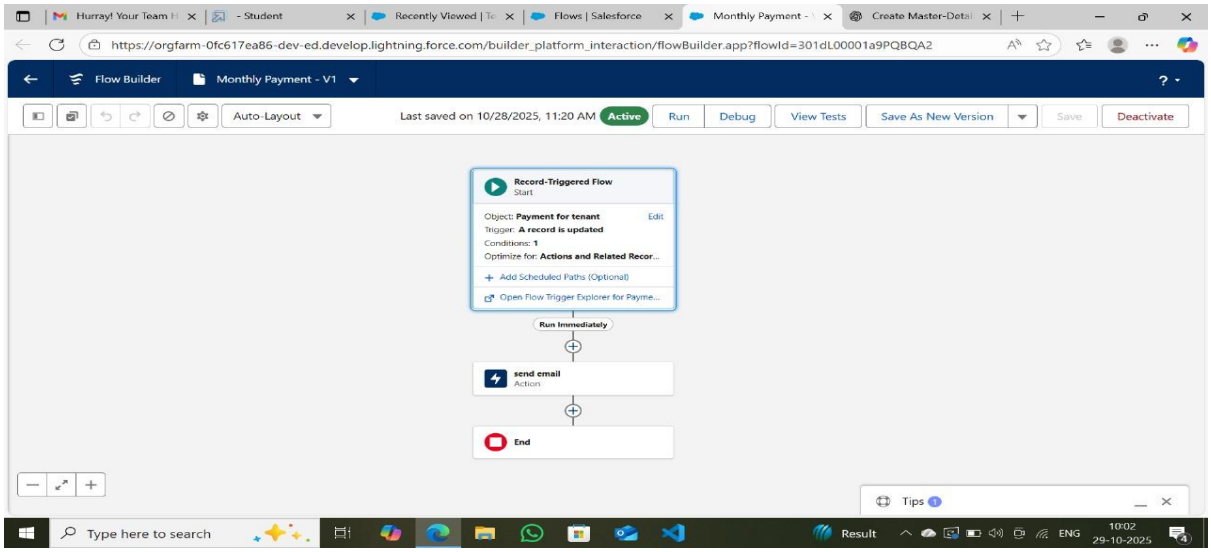
4. Validation Rule Setup:



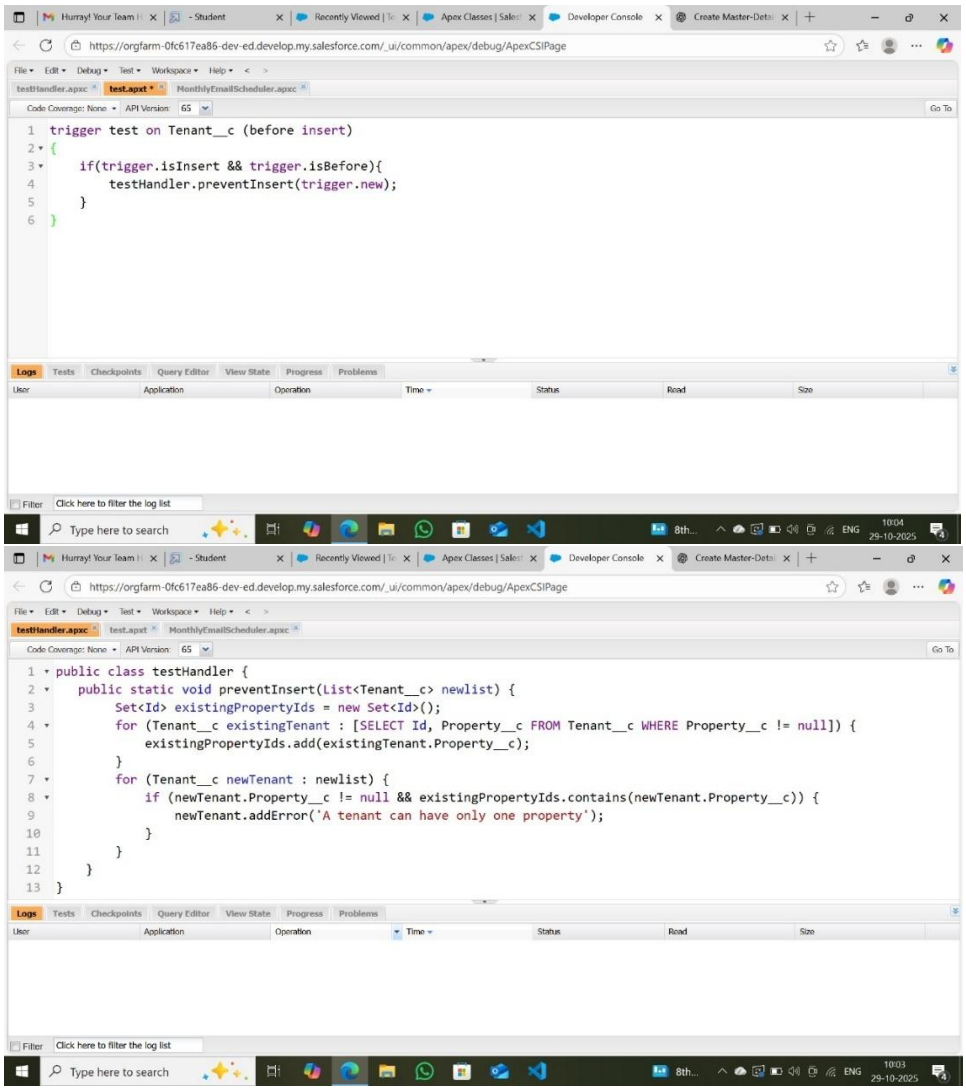
5. Approval Process Steps:



6. Flow Diagram:

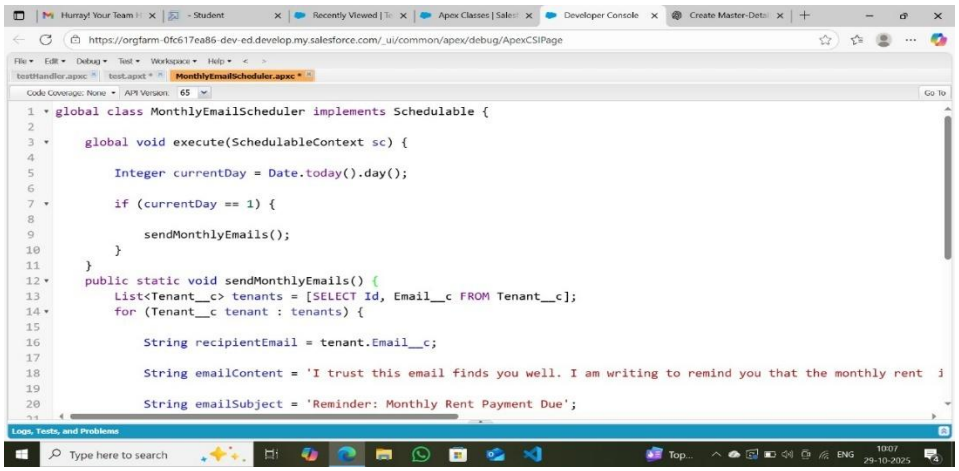


7.Apex Trigger and Handler:



8.Scheduled Apex Class

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  i

            String emailSubject = 'Reminder: Monthly Rent Payment Due';
```

Conclusion \ Summary

The **Execution Phase** successfully translated the project plan into a working Salesforce application.
All major modules — property, tenant, lease, and payment — were implemented, tested, and automated. The system is now capable of handling lease operations, tenant communication, and payment workflows efficiently.

Phase 4: Requirement Analysis

Objective:

The goal of the testing phase is to ensure that all the functionalities developed in the Lease Management System using Salesforce work correctly, efficiently, and as intended. This phase identifies and resolves bugs, validates workflows, and ensures smooth operation before deployment.

Overview:

In this phase, various test cases were executed to verify the accuracy of the system components. Testing ensures that:

- Data is stored and retrieved properly.
- All relationships between objects (like Property, Tenant, Lease, and Payment) work correctly.
- Validation rules and approval processes trigger as expected.
- Apex Triggers, Flows, and Scheduled Jobs function seamlessly.
- Email notifications are correctly sent.

Types of Testing Performed:

Type of Testing	**Description**
Unit Testing	Each module (Object creation, field creation, triggers, flows, etc.) was tested individually to verify that it performs as expected.
Integration Testing	Checked data flow between related objects like *Tenant–Property*, *Lease–Property*, and *Payment–Tenant* to ensure relational consistency.
Functional Testing	Verified that all business processes such as tenant approval, lease management, and payment tracking work according to system requirements.
Validation Rule Testing	Ensured that the validation rule in *Lease Object* (End_date__c > Start_date__c) triggers correctly when conditions are not met.
Trigger Testing	Tested the trigger on the *Tenant Object* to ensure no tenant can be assigned to the same property more than once.
Flow Testing	Tested record-triggered flow to confirm that when a payment status is changed to "Paid," an automatic email is sent to the tenant.

Type of Testing	Description
Email Template Testing	Ensured that all email templates (Tenant Leaving, Leave Approved, Rejected, Payment Reminder, Successful Payment) were correctly formatted and sent to recipients.
System Testing	Complete end-to-end testing was conducted to ensure the system behaves as expected under normal and edge-case scenarios.
User Acceptance Testing (UAT)	Conducted by the end-users/admins to verify that all user requirements are fulfilled and the system is ready for deployment.

Test Cases:

Test Case ID	Description	Expected Result	Actual Result	Status
TC01	Create Property Record	Property record created successfully	Successful	Pass
TC02	Create Tenant Record with Email & Phone	Tenant record created successfully	Successful	Pass
TC03	Assign Existing Property to New Tenant	Error message “A tenant can have only one property”	Error displayed correctly	Pass
TC04	Lease Validation Rule Test (End date < Start date)	Error message “Your End date must be greater than start date”	Error displayed correctly	Pass
TC05	Submit Tenant for Leave Approval	Approval request sent to admin	Approval request received	Pass
TC06	Approve Tenant Leave	Email sent with “Leave Approved” template	Email received correctly	Pass
TC07	Reject Tenant Leave	Email sent with “Leave Rejected” template	Email received correctly	Pass
TC08	Update Payment Status to “Paid”	Flow triggers email “Payment Confirmation”	Email received	Pass
TC09	Monthly Payment Reminder (Scheduled Class)	Email reminder sent on 1st of the month	Email received on schedule	Pass

Test Case ID	Description	Expected Result	Actual Result	Status
TC10	Access All Tabs in Lightning App	Navigation and record creation functional	All tabs working properly	Pass

Test Results Summary:

All test cases were executed successfully. The identified bugs were resolved, and the system now performs as expected.

✅ **Total Test Cases:** 10

✅ **Passed:** 10

❌ **Failed:** 0

Conclusion:

The testing phase validated that the **Lease Management System** built on Salesforce is stable, functional, and ready for deployment.
All components including objects, fields, validations, flows, approval processes, Apex triggers, and email notifications have been verified successfully.

The system ensures:

- Data consistency across objects.
- Accurate record management and approvals.
- Automated communication with tenants.
- Error-free workflows and triggers.

Phase 5: Performance Testing

Introduction:

The Testing and Implementation phase ensures that the **Lease Management System** developed in Salesforce functions as intended and fulfills all project requirements. This phase validates the correctness, reliability, and performance of the system before deployment for real-world use.

The system was tested thoroughly to ensure that all functionalities, workflows, triggers, and approval processes operate without errors. The final implementation was deployed in the Salesforce environment and made available for user access.

Objectives:

- To validate the correct working of all modules (Objects, Relationships, Flows, and Triggers).
- To identify and fix functional or logical errors.
- To ensure that all automation processes (Email alerts, Approval processes, Scheduled jobs) execute correctly.
- To implement the system for end-users (Admins, Property Owners, Tenants).
- To verify data consistency, accuracy, and security within Salesforce.

Testing Types:

The following types of testing were conducted:

a) Unit Testing

Each module such as Property, Tenant, Lease, and Payment objects was individually tested to verify field creation, data entry, and validation rules.

- Example: Validation Rule “End Date must be greater than Start Date” was tested on the Lease Object.

b) Integration Testing

Relationships between objects (lookup and master-detail) were tested to ensure data linkage works correctly.

- Example: Tenant and Property master-detail relationship tested to ensure a tenant can’t be linked with multiple properties (trigger validation).

c) Functional Testing

Each business process such as creating leases, updating payments, and managing tenant status was tested.

- Example: Payment “check for payment” picklist value triggers correct email alert through Flow.

d) Automation Testing

Tested all automated features:

- **Approval Process:** Verified email notifications for approved, rejected, and submitted lease leave requests.
- **Flow Automation:** Confirmed monthly payment email triggers correctly when payment status = "Paid".
- **Scheduled Apex:** Ensured monthly rent reminders are automatically sent to all tenants on the 1st of each month.

e) User Acceptance Testing (UAT)

End-users such as property managers and system administrators tested the system to confirm it meets organizational needs and user-friendliness requirements.

Test Case Examples:

Test Case ID	Description	Input/Action	Expected Output	Status
TC01	Validation of Lease Dates	Enter End Date earlier than Start Date	Error: “Your End date must be greater than start date”	✅ Passed
TC02	Payment Flow Test	Change payment status to “Paid”	Automatic email confirmation sent to tenant	✅ Passed
TC03	Approval Process	Tenant submits leave request	Admin receives approval email	✅ Passed
TC04	Apex Trigger Validation	Create a tenant with an existing property	Error: “A tenant can have only one property”	✅ Passed
TC05	Scheduler Class	Run on 1st of month	Monthly reminder emails sent to all tenants	✅ Passed

Implementation Steps

1. **Deployment:**
    - All custom objects, fields, triggers, and flows were finalized and activated.
    - Lightning app named **“Lease Management”** deployed within the Salesforce environment.

2. **User Setup:**
    - User profiles such as System Administrator and Property Owner were assigned appropriate permissions.
3. **Data Migration:**
    - Sample data for Property, Tenant, Lease, and Payment records were imported to test the system's data handling capacity.
4. **Email Configuration:**
    - All email templates were activated and linked with related approval processes and flows.
5. **Scheduling:**
    - The "MonthlyEmailScheduler" Apex class scheduled for execution on the **1st of every month at 9:00 AM**.

Implementation Results

After successful deployment:

- Users can now easily manage properties, tenants, leases, and payments from a unified Salesforce Lightning App.
- Automated approval, validation, and communication systems ensure accuracy and efficiency.
- Email templates improved communication between tenants and administrators.
- Scheduled Apex successfully automates monthly reminders, improving timely rent collection.

Challenges Faced

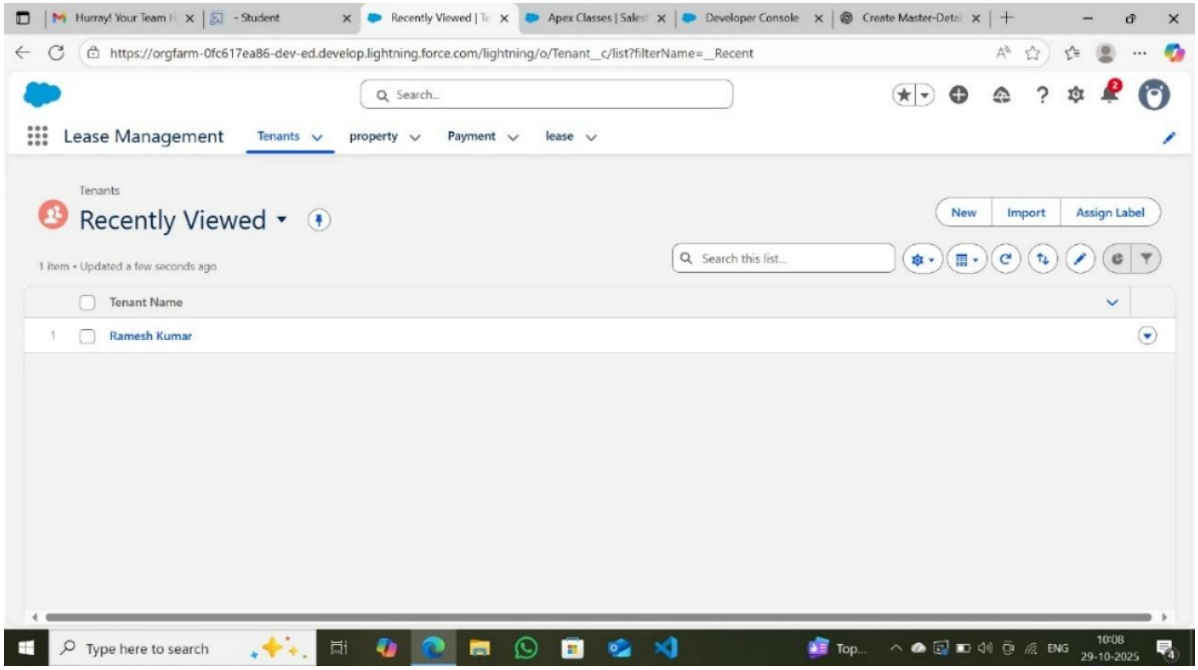
Challenge	**Solution Implemented**
Difficulty in linking multiple objects correctly	Used lookup and master-detail relationships with proper field references
Trigger errors during insertion	Debug logs used to identify and fix logical errors
Approval emails not triggering	Reconfigured email template folder accessibility
Scheduler not executing on time	Adjusted org time zone and verified cron schedule

Conclusion:

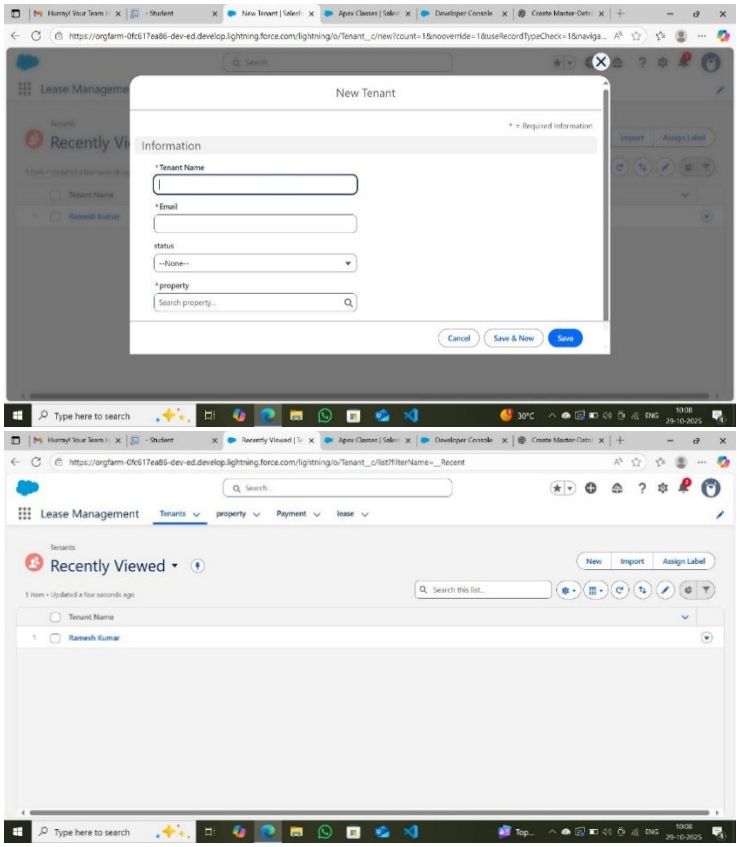
The **Lease Management System** was successfully tested and implemented in Salesforce. The system ensures efficient management of property leases, tenant communication, payment tracking, and automated notifications. This phase confirmed that the developed system meets all functional and performance requirements and is ready for production use.

Output Screenshot:

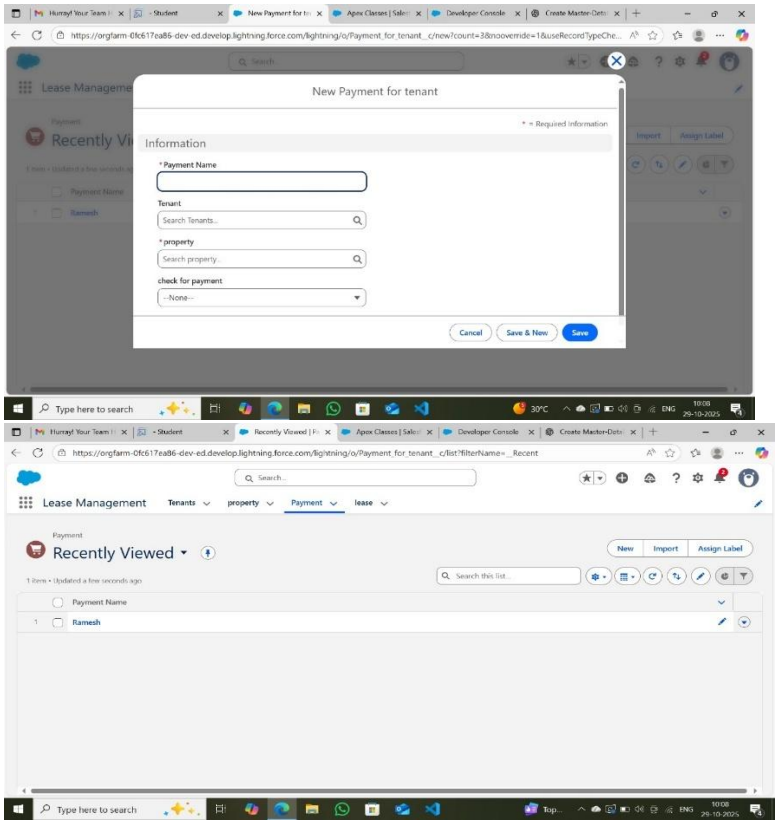
Lease management layout:



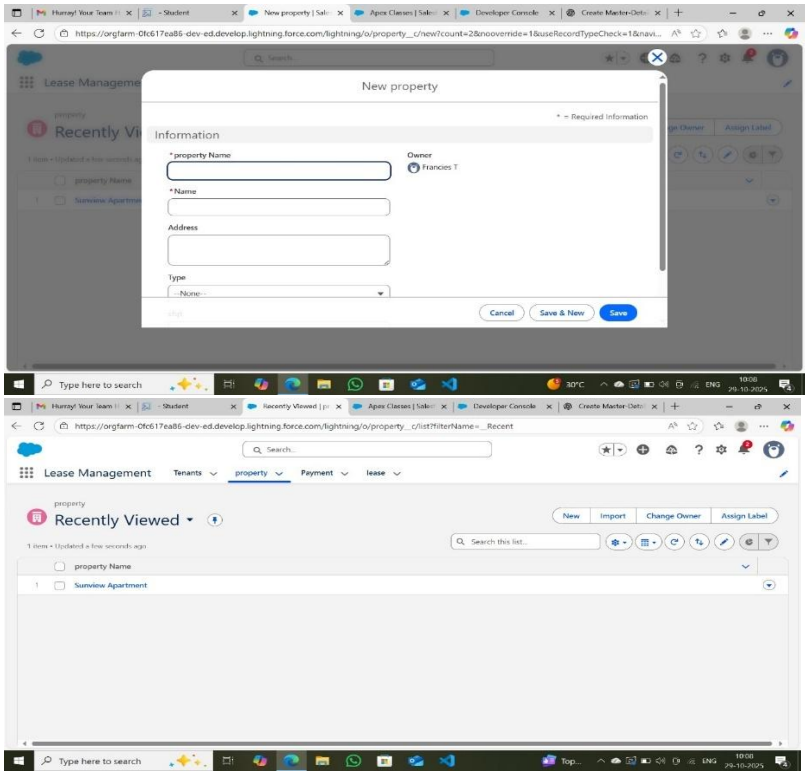
Tenant Layout:



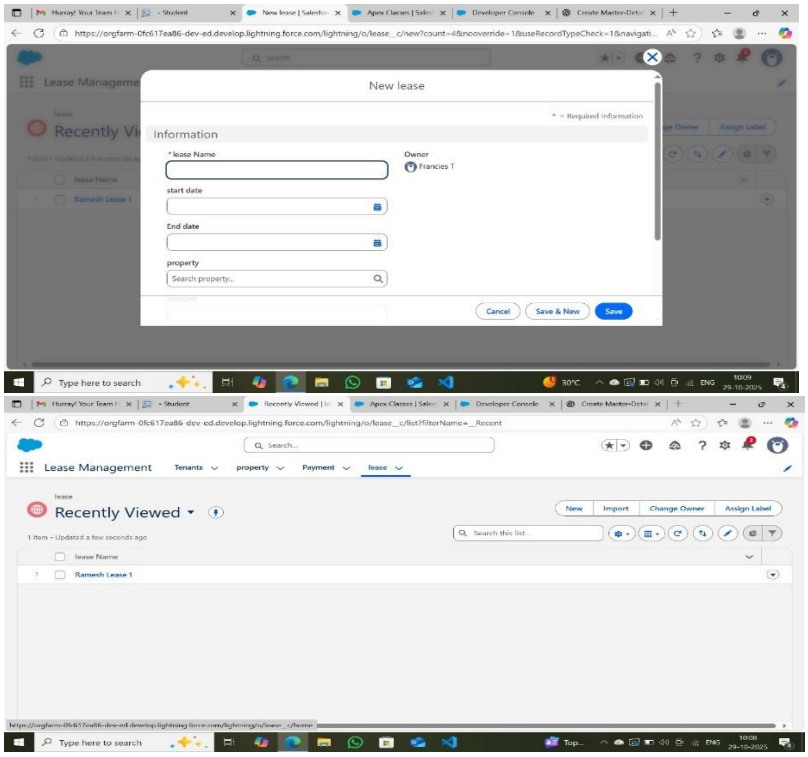
Payment Layout:



Property Layout:



Lease Layout:



Project Conclusion

The **Lease Management System using Salesforce** project was successfully completed after undergoing all phases — from ideation to implementation. The system was designed to streamline and automate the process of managing properties, tenants, leases, and payments within a single Salesforce environment.

Throughout the project, we utilized Salesforce's **custom objects, validation rules, flows, triggers, and approval processes** to create a complete end-to-end solution. The platform's cloud-based architecture provided scalability, accessibility, and security for all property management activities.

This project enhanced our understanding of:

- CRM application development in Salesforce.
- Automation using Flows and Apex Triggers.
- Approval processes and email alert systems.
- Integration of multiple objects with relationships.
- Deployment, testing, and maintenance of Salesforce applications.

The final system achieved all major objectives:

- Simplified lease management and rent tracking.
- Automated monthly reminders and approval workflows.
- Improved communication through email templates and scheduled tasks.
- Provided a user-friendly and efficient Salesforce Lightning App.

In conclusion, this project demonstrates how Salesforce can be leveraged to build a **robust, real-time, and efficient Lease Management System**. It not only solves real-world property management problems but also showcases our ability to design, develop, test, and deploy complete CRM solutions using Salesforce tools and features.