

# **SHIVANI ENGINEERING COLLEGE**

Department of Computer science and engineering

## **PROJECT TITLE:**

## **LEASE MANAGEMENT SYSTEM**

### **Submitted by (Team Members):**

Team ID: NM2025TMID00660

Team Size: 4

Team Leader: Swetha M

Team member: Amsavalli M

Team member: Yuvarani A

Team member: Saranya A

Guide by: Archana N

**Platform:** Salesforce Developer

**Date of Submission:** 04/11/2025

## Project Description

A lease management project involves creating a system or application to efficiently handle the processes related to leasing real estate properties, equipment, or other assets.

The goal is to streamline and automate various tasks associated with lease agreements, ensuring accurate record-keeping, compliance with regulations, and effective communication between parties involved.

## Creating Developer Account

In this step, a new Salesforce Developer Account is created using a valid email address. This account provides access to the development environment for project setup.

## Account Activation

Activate the Salesforce Developer Account by verifying the registered email and completing the activation process.

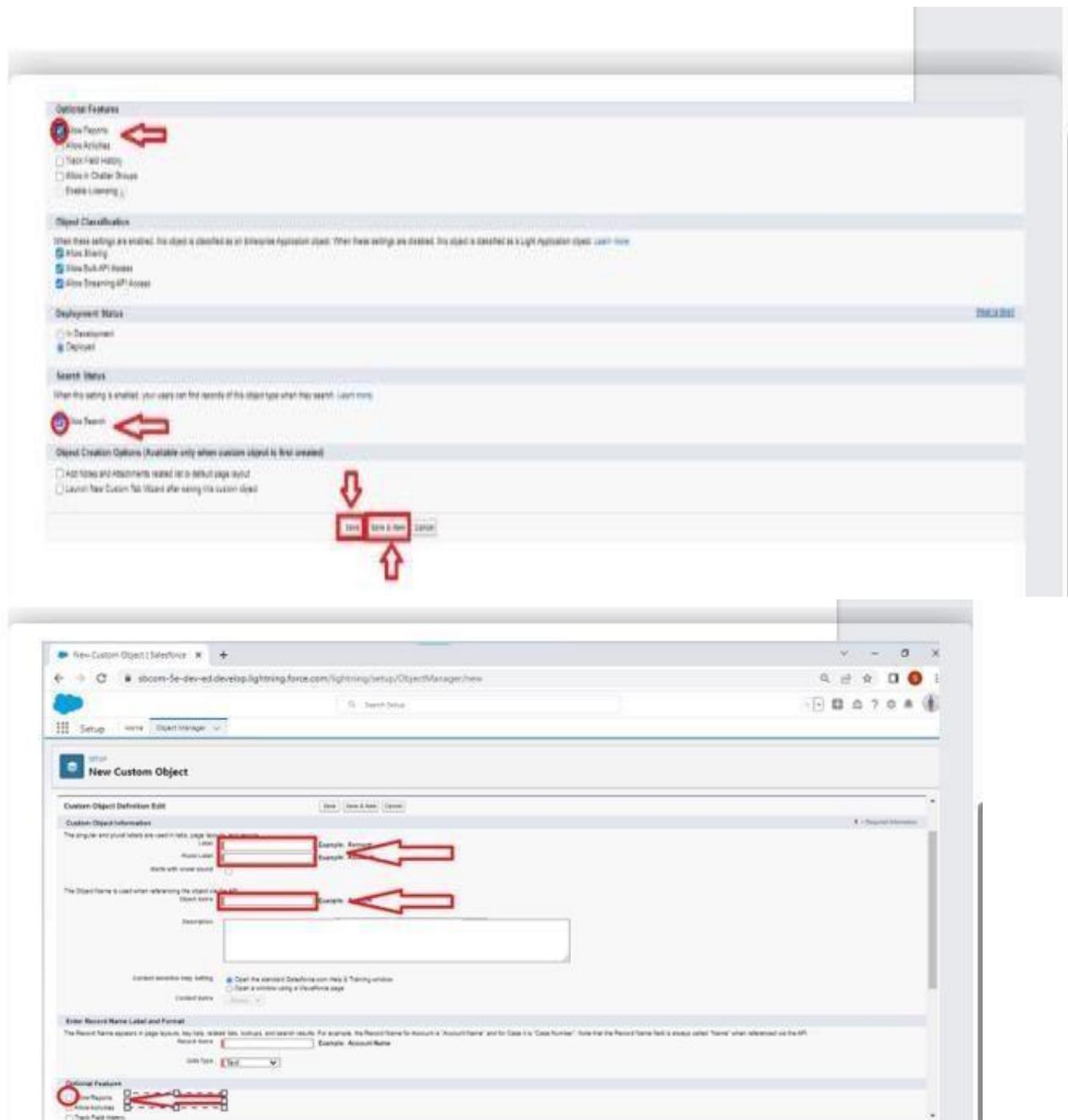
The screenshot shows the Salesforce Setup Home page. On the left, there's a sidebar with links like Setup Home, Service Setup Assistant, Multi-factor Authentication Assistant, Release Updates, Lightning Experience Transition Assistant, Salesforce Mobile App, Lightning Usage, Optimizer, and Administration. The main area features three cards: "Get Started with Einstein Bots" (Launch an AI-powered bot to automate your digital connections), "Mobile Publisher" (Use the Mobile Publisher to create your own branded mobile app), and "Real-time Collaborative Docs" (Transform productivity with collaborative docs, spreadsheets, and slides inside Salesforce). A "Create" button is in the top right.

The screenshot shows the "Sign up for your Salesforce Developer Edition" page. It features a large image of a computer monitor displaying a complex application interface. Below it, the text reads: "Build enterprise-quality apps fast to bring your ideas to life". A bulleted list includes: "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". To the right, there's a form to sign up with fields for First Name, Last Name, Email, and Role, along with a Company field. A note says: "Complete the form to start your free trial. Our team will be in touch to help you make the most of your trial."

## Create Property Object

Create a new custom object named “Property” to store details related to lease properties such as property name, type, and location.



## Create Tenant Object

Create a “Tenant” custom object to manage tenant information including name, contact, and lease duration.

## Create Payment Object

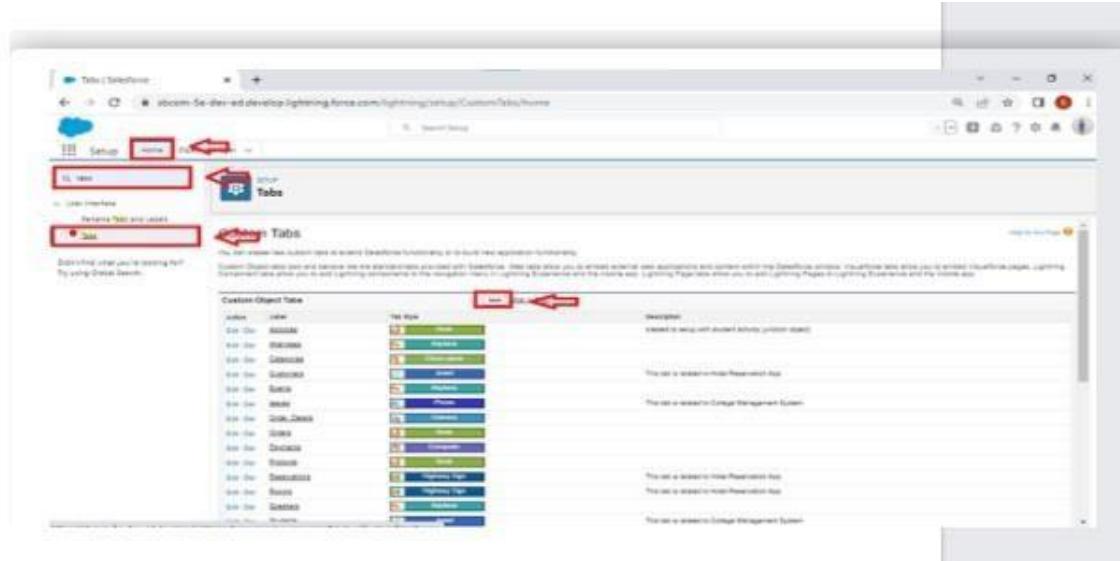
Design a “Payment” object to handle rent payments, due dates, and payment status tracking.

## Create Lease Object

Establish a “Lease” object to connect Property and Tenant objects and manage lease terms, start and end dates.

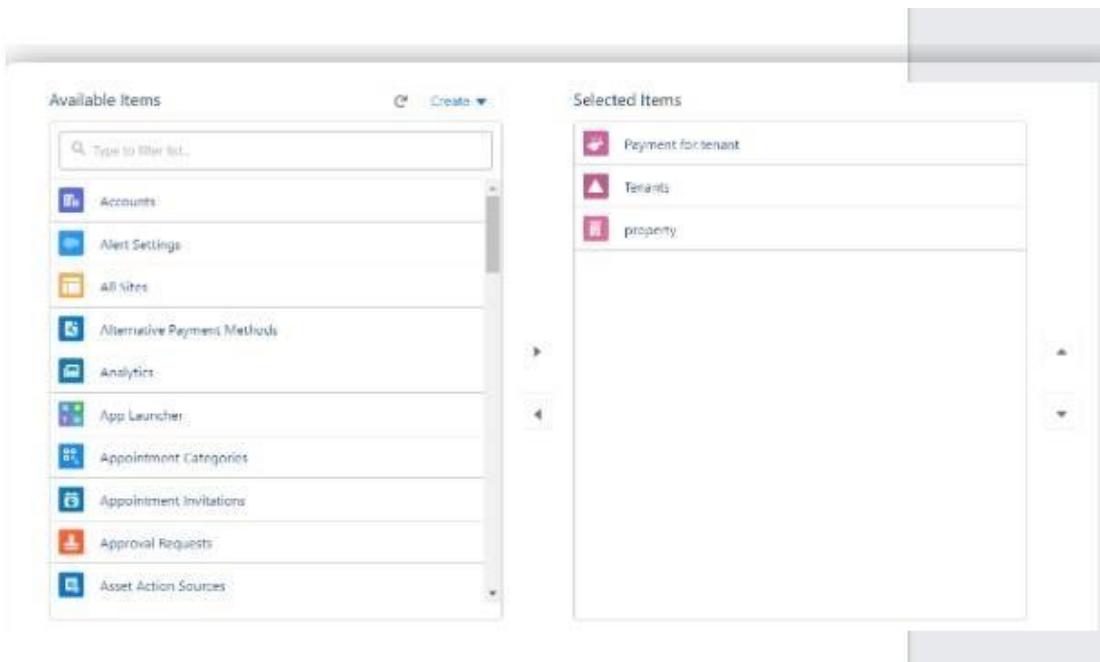
## Creating a Custom Tab

Add custom tabs for each created object to make them accessible in the Salesforce app navigation bar.



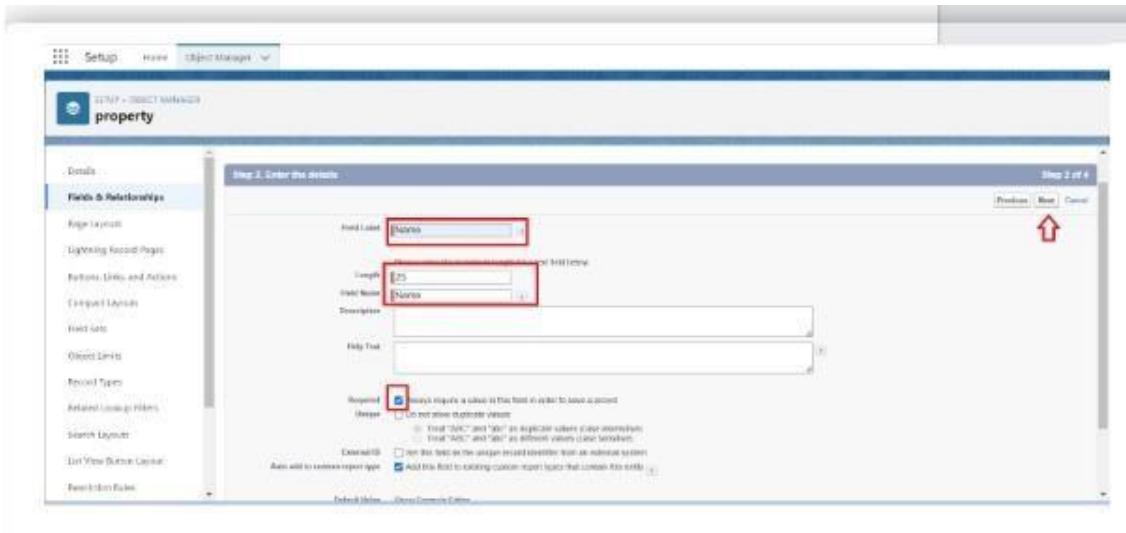
## Lightning App Creation

Create a custom Lightning App that includes all the custom objects and tabs for better project navigation. **Fields Creation**



## Fields Creation

In Salesforce, **Fields** are used to store different types of data inside each **Object**. Each object in the **Lease Management System** has specific fields that help to manage properties, tenants, leases, and payments efficiently.



## Email Template

An **Email Template** in Salesforce is a pre-designed message format that allows users to send standardized emails automatically or manually. It saves time, ensures consistency, and helps communicate important information like **lease confirmation, payment receipts, and reminders** to tenants.

### 1. Tenant Leaving Notification

Sends an alert to the admin when a tenant requests to vacate the property.

### 2. Lease Approved

Notifies the tenant that their lease request has been approved successfully.

### 3. Lease Rejection Email

Informs the tenant that their lease application has been rejected due to missing details or eligibility.

### 4 . Monthly Payment Reminder

Automatically reminds the tenant each month about the upcoming rent payment due date.

### 5. Successful Payment Confirmation

Sends a thank-you message and confirmation once the tenant's rent payment is received successfully.

## Approval Process

An **Approval Process** automates how records are approved in Salesforce. In this project, an approval process is created for the **Lease or Payment** object to ensure that certain records (for example, rent amount > ₹50,000) need admin or manager approval before final confirmation.



## Apex Trigger

An **Apex Trigger** is used to perform automatic actions when a record is created or updated. In this project, a trigger is written on the **Payment** object to automatically update the **Lease Status** once the payment is completed.

```

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

```

## Apex logic:

```

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) {

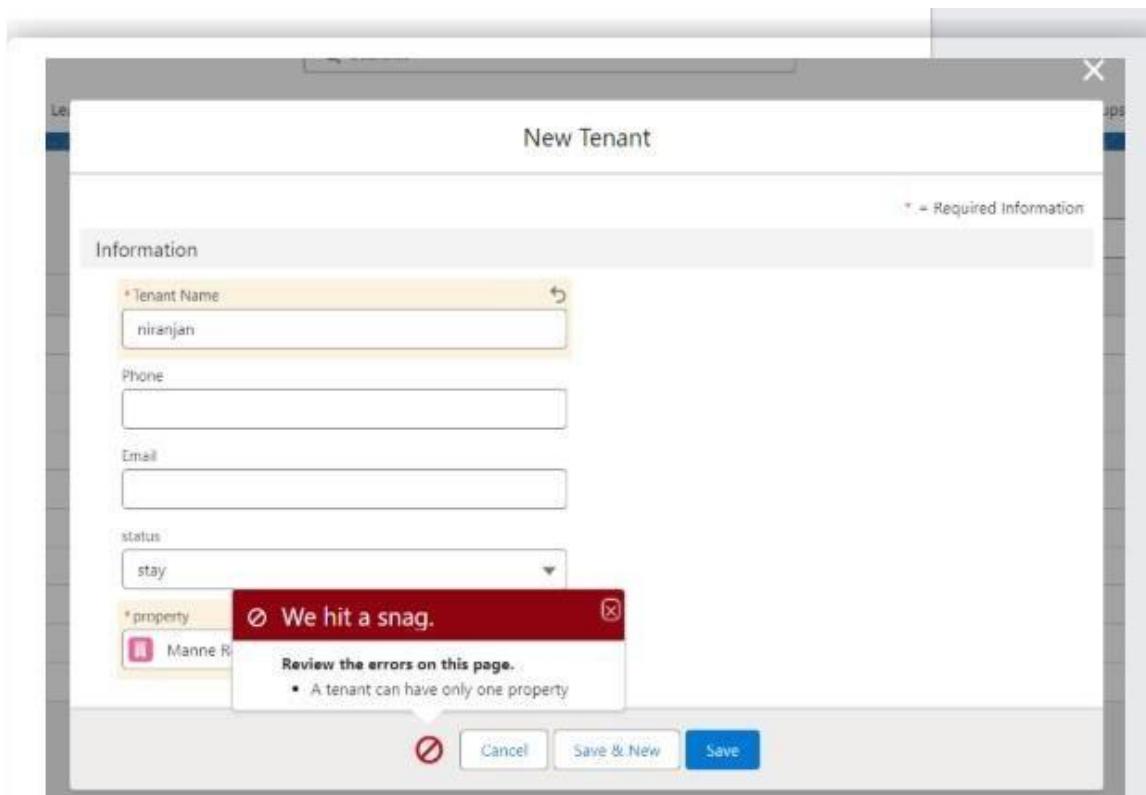
```

```

        if (newTenant.Property__c != null &&
existingPropertyIds.contains(newTenant.Property__c)) {
            newTenant.addError('A tenant can have only one property');
        }
    }
}
}

```

## Testing the Trigger:



## Flow Creation

A **Flow** is created to automate record updates or send email alerts without coding.

In this project, a **Record-Triggered Flow** is designed on the **Tenant** or **Payment** object to send an automatic email when a new payment is recorded.

**Set Entry Conditions**

Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.

If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the **Only when a record is updated to meet the condition requirements** option for When to Run the Flow for Updated Records.

**Condition Requirements:**

All Conditions Are Met (AND)

Field	Operator	Value
check_for_payment_c	Equals	paid

+ Add Condition

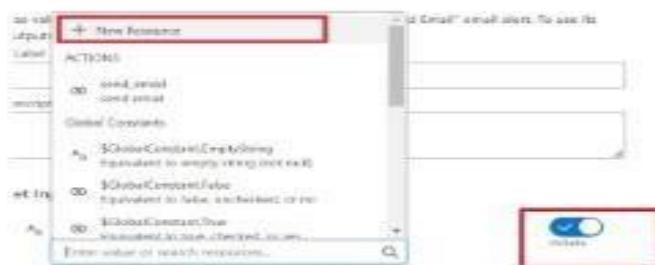
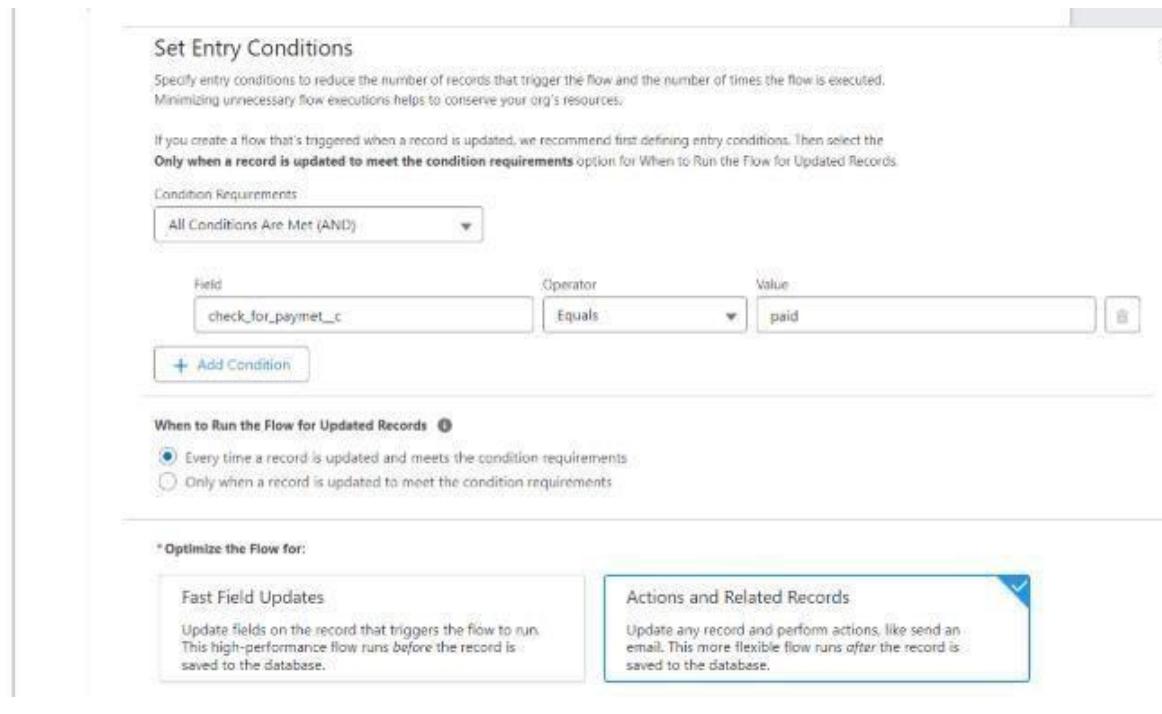
**When to Run the Flow for Updated Records:**

Every time a record is updated and meets the condition requirements  
 Only when a record is updated to meet the condition requirements

**\* Optimize the Flow for:**

Fast Field Updates: Update fields on the record that triggers the flow to run. This high-performance flow runs before the record is saved to the database.

Actions and Related Records: Update any record and perform actions, like send an email. This more flexible flow runs after the record is saved to the database.



## Schedule Apex Class

A **Scheduled Apex Class** is used to run Apex code automatically at a specific time (daily, weekly, monthly). For example — sending payment reminders, lease expiry alerts, or autoupdating records every morning.

Schedule Apex

Schedule an Apex class that implements the 'Schedulable' interface to be automatically executed on a weekly or monthly interval.

Job Name:	MonthlyEmailScheduler	Save   Cancel
Apex Class:	MonthlyEmailScheduler	<input type="button" value="New"/>
Schedule Apex Execution		
Frequency:	<input type="radio"/> Weekly <input checked="" type="radio"/> Monthly	<input checked="" type="radio"/> On May 1 <input type="checkbox"/> of every month <input type="radio"/> On Jun 1 <input type="checkbox"/> of every month <input type="radio"/> On Jul 1 <input type="checkbox"/> of every month
Start:	04/12/2023	14:00:00 UTC
End:	04/01/2024	14:00:00 UTC
Preferred Start Time:	9:00 am	
Each start time will depend on job queue activity.		
Save   Cancel		

The screenshot shows the Apex Classes page in the Salesforce Lightning Experience. A red box highlights the 'Delete' button in the top right corner of the table header. Another red box highlights the 'New Class' button in the top left corner of the page.

**Apex Classes**

Apex Classes is an object-oriented programming language that allows developers to create custom business logic in their applications on the Lightning Platform.

Recent Apex Classes (Last 12 hours)

Customize this page

Filter: All Classes

Name	Class Type	Managed Package	Created From API	Run All Tests	Delete
Bill Due Monthly	Standard	001	Active	100	<input type="button" value="Delete"/>
Bill Due Monthly	Standard	002	Active	100	<input type="button" value="Delete"/>
Bill Due Monthly	Standard	003	Active	100	<input type="button" value="Delete"/>
Bill Due Monthly	Standard	004	Active	100	<input type="button" value="Delete"/>

## Conclusion

The *Lease Management System* project was successfully created using Salesforce. This project demonstrates how different Salesforce features such as **Objects, Tabs, Validation Rules, Email Templates, Approval Process, Flows, and Apex Triggers** can be used to automate and simplify lease management tasks. The system ensures efficient handling of tenant records, payments, and lease approvals with minimal manual work. Overall, this project improves productivity, reduces human error, and provides a digital solution for real-time property management.

