Brainstorming & Ideation

1. Define Problem Statements

Managing lease agreements manually can be a complex and time-consuming process for both property owners and tenants. Many organizations still rely on spreadsheets or paper-based systems to track property details, tenant information, lease periods, and payment schedules. This manual approach often leads to data inconsistency, missed deadlines, and difficulty in accessing vital records.

The **Lease Management System** aims to overcome these challenges by developing a **centralized Salesforce-based solution** that automates and streamlines the entire leasing process. Using Salesforce allows seamless integration of data, efficient record-keeping, timely communication, and enhanced visibility into all lease-related activities.

Key Problems Identified:

- Difficulty in maintaining accurate lease records across multiple properties.
- Missed payment reminders and renewal deadlines due to lack of automation.
- Lack of centralized access to tenant and payment information.
- Inability to track vacant or expired leases effectively.
- Limited communication between tenants and property managers.
- No structured approval or notification system for lease processes.

Objective:

To design and develop a **Salesforce-based Lease Management System** that manages property, tenant, lease, and payment information efficiently, automates notifications, and supports approval workflows for smoother operations.

2. Empathy Map

WHO:
Property owners, tenants, and lease managers

THINK & FEEL:
“I need a simple, reliable way to manage leases and payments.”

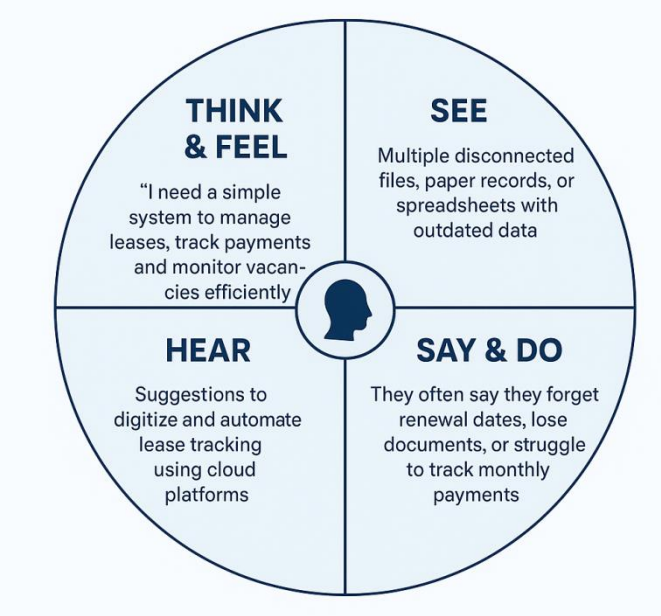
SEE:
Disorganized spreadsheets and outdated data

SAY & DO:
“I forget renewal dates and lose lease documents.”

HEAR:
“Try automating lease tracking with Salesforce.”

PAIN POINTS:
Missed payments, document loss, manual approval delays

GAINS:
Centralized data, auto-reminders, reports, and secure cloud storage



3. Brainstorming

During the brainstorming sessions, various ideas were discussed to enhance the lease management process and determine the best approach for implementation on the Salesforce platform. The focus was on automating manual processes and providing transparency between tenants and owners.

Ideas Generated:

1. Develop a **Salesforce Lightning App** for lease management.
2. Create **custom objects** for Property, Tenant, Lease, and Payment.
3. Use **validation rules** to ensure data accuracy in lease entries.

4. Implement **approval processes** for vacant checks and lease renewals.
5. Design **email templates** for automated communication such as payment reminders, approvals, and tenant notifications.
6. Create **Apex triggers** and **flows** for automating monthly payment reminders.
7. Use **lookup fields** to interlink related records between Property, Tenant, and Lease objects.
8. Schedule **Apex classes** for periodic tasks such as updating lease statuses.

Selected Solution: A **custom Salesforce Lightning App** named *Lease Management* that integrates all lease-related modules — properties, tenants, payments, and lease details — under one unified platform with automation and approval processes.

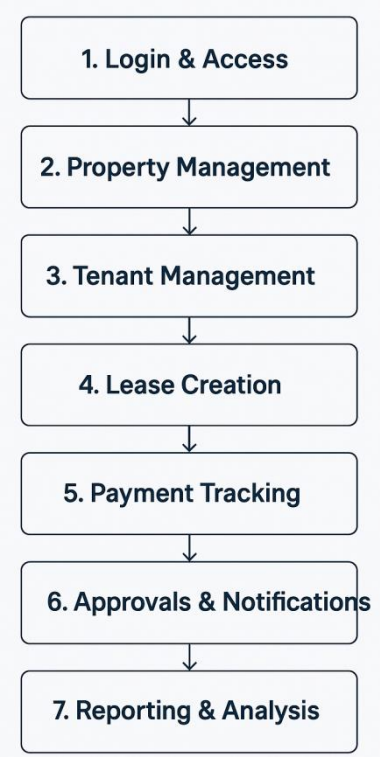
Requirement Analysis

1. Customer Journey Map

The customer journey for the **Lease Management System** outlines the step-by-step experience of both property owners and tenants using the Salesforce platform. It focuses on how users interact with the system — from login to managing lease operations — ensuring convenience, automation, and transparency throughout the process.

Stage	User Action	System Interaction (Salesforce)	Output/Experience
1. Login & Access	The user logs in using Salesforce credentials.	Salesforce authenticates and directs the user to the *Lease Management Lightning App*.	Secure access to the dashboard.
2. Property Management	The admin adds new property details.	Data stored in the **Property Object**.	Property records become viewable and editable.
3. Tenant Management	The user registers tenant details.	Stored in the **Tenant Object** linked to Property.	Tenant data displayed with lease association.

4. Lease Creation	The admin creates a new lease agreement.	Record stored in the **Lease Object** with validation rules.	Lease data automatically connected to property and tenant.
5. Payment Tracking	Monthly payments are logged or auto-reminded.	Managed through **Payment Object**, triggers, and flows.	Email notifications sent for payment status.
6. Approvals & Notifications	Approvals triggered for lease renewals or vacant checks.	Approval process workflow initiated in Salesforce.	Approval status updated automatically.
7. Reporting & Analysis	Admin views analytics and reports.	Salesforce dashboard and reports visualize lease data.	Insights into rent collection, active leases, and vacancies.



This journey ensures **ease of navigation, automation, and efficient record management**, improving both user satisfaction and operational efficiency.

2. Solution Requirement

The **Lease Management System** requires a combination of functional and non-functional features to achieve full automation and user satisfaction.

Functional Requirements:

1. Ability to add, edit, and delete property and tenant details.
2. Create and manage lease records with automatic start and end date validation.
3. Generate reminders for lease renewals and rent payments.
4. Enable automated email communication for approvals, rejections, and payments.
5. Support an approval process for checking vacant properties.
6. Maintain relationships between objects through lookup fields (Property ↔ Tenant ↔ Lease).
7. Provide dashboards and reports for monitoring lease performance.

Non-Functional Requirements:

1. **Scalability** – The system should support multiple properties and tenants.
2. **Security** – Only authorized users should access or modify records.
3. **Performance** – Quick response to data queries and reports.
4. **Reliability** – Data must remain consistent even during workflow automation.
5. **Usability** – A user-friendly Lightning interface for both admins and tenants.

3. Data Flow Diagram (DFD)

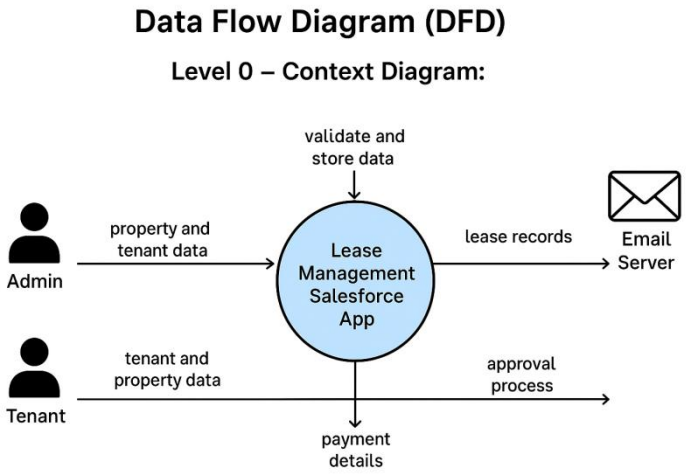
Level 0 – Context Diagram:

- **Users:** Admin, Tenant
- **System:** Lease Management Salesforce App
- **External Entities:** Email Server (for notifications)

Data Flow:

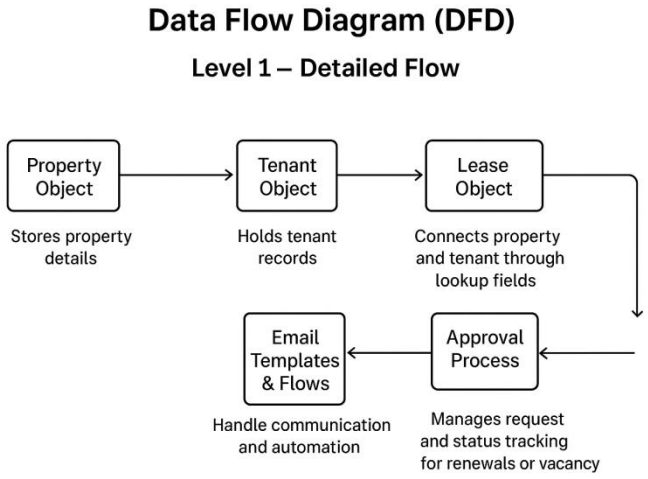
1. Admin enters property and tenant data into Salesforce.
2. System validates and stores data in the respective custom objects.
3. Lease records are created linking tenant and property.

4. Payment details are recorded and notifications are triggered via email.
5. Approval process initiated for vacancy checks or renewals.
6. Reports generated for management overview.



Level 1 – Detailed Flow:

- **Property Object** → Stores property details.
- **Tenant Object** → Holds tenant records.
- **Lease Object** → Connects property and tenant through lookup fields.
- **Payment Object** → Tracks payments and schedules automated reminders.
- **Email Templates & Flows** → Handle communication and automation.
- **Approval Process** → Manages request and status tracking for renewals or vacancy.



This ensures smooth flow of data among modules while maintaining consistency and automation.

4. Technology Stack

Component	Technology Used	Purpose
Platform	Salesforce Lightning	To build and deploy cloud-based applications.
Database	Salesforce Object Database (Custom Objects)	To store property, tenant, lease, and payment data.
Backend Logic	Apex Classes, Apex Triggers	To automate processes and enforce business logic.
Frontend/UI	Lightning Components, Tabs, and Pages	For user interface design and navigation.
Automation	Flows, Validation Rules, Approval Processes	To manage workflows and system interactions.
Communication	Email Templates, Email Alerts	To send automatic notifications to users.
Scheduling	Scheduled Apex Classes	To handle periodic actions such as monthly payments.
Security & Access	Role-based Permissions in Salesforce	To ensure data privacy and restricted access.

Project Design Phase

1. Problem–Solution Fit

Managing lease agreements traditionally involves scattered data across spreadsheets or paper records. This leads to difficulties in tracking payments, lease renewals, and vacant properties, often causing financial loss and miscommunication.

To solve these problems, **Salesforce**, a robust CRM and automation platform, is used to create a centralized, automated, and easily accessible **Lease Management System**.

Key Problem Areas vs. Solutions:

Problem	Proposed Solution (Using Salesforce)
Manual tracking of lease details	Create custom **objects** (Property, Tenant, Lease, Payment) to store and manage lease data systematically.
Missed payments and renewals	Use **Flows** and **Scheduled Apex Classes** to send reminders automatically.
Lack of communication between owner and tenant	Build **Email Templates** and **Email Alerts** for notifications.
Data inconsistency	Apply **Validation Rules** and **Lookup Fields** to ensure relational integrity.
No approval or review process	Design an **Approval Process** for vacant property checks and renewals.
Limited insights into operations	Generate **Salesforce Reports and Dashboards** for visualization and decision-making.

2. Proposed Solution

The proposed **Lease Management System** is a Salesforce-based cloud application designed to streamline and automate the leasing process for property owners and tenants.

Core Features:

1. **Centralized Data Storage:**
    - Custom Objects for *Property*, *Tenant*, *Lease*, and *Payment*.
    - Lookup relationships to maintain data linkage.

2. **Automation:**
    - Flows for monthly rent reminders.
    - Scheduled Apex for recurring notifications.
    - Apex Triggers to automate dependent record updates.
3. **Communication:**
    - Predefined Email Templates for approvals, payments, and tenant updates.
    - Email Alerts for both successful and rejected transactions.
4. **Validation & Approval:**
    - Validation rules to prevent incomplete or incorrect data.
    - Approval process for verifying lease renewals and vacant property checks.
5. **Analytics & Reporting:**
    - Real-time dashboards to monitor lease status, payments, and property occupancy.
6. **User Interface (UI):**
    - Custom **Lightning App** for easy navigation and management.
    - Tabs for each object and module for structured access.

3. Solution Architecture

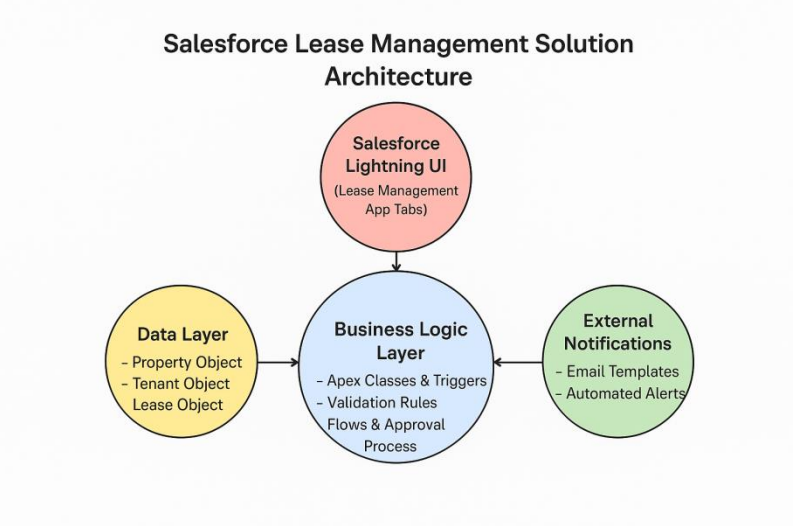
The solution architecture defines how various Salesforce components interact to deliver the complete lease management workflow.

Architecture Layers:

- **Presentation Layer:**
    - Salesforce Lightning App (user interface).
    - Tabs for Property, Tenant, Lease, and Payment management.
- **Business Logic Layer:**
    - Apex Triggers and Classes to handle system logic (e.g., auto-update payment status).
    - Validation Rules and Approval Processes for maintaining data integrity.
    - Flows for automated monthly notifications.
- **Data Layer:**
    - Custom Objects for Property, Tenant, Lease, and Payment.
    - Lookup relationships linking all relevant records.

Data Interaction Flow:

1. Admin creates **Property** and **Tenant** records.
2. **Lease** is created linking both entities.
3. **Payment** records are generated monthly via Flows or Apex Classes.
4. **Email Templates** notify tenants about payments or approvals.
5. **Reports & Dashboards** summarize all active leases and payments.



Project Planning Phase

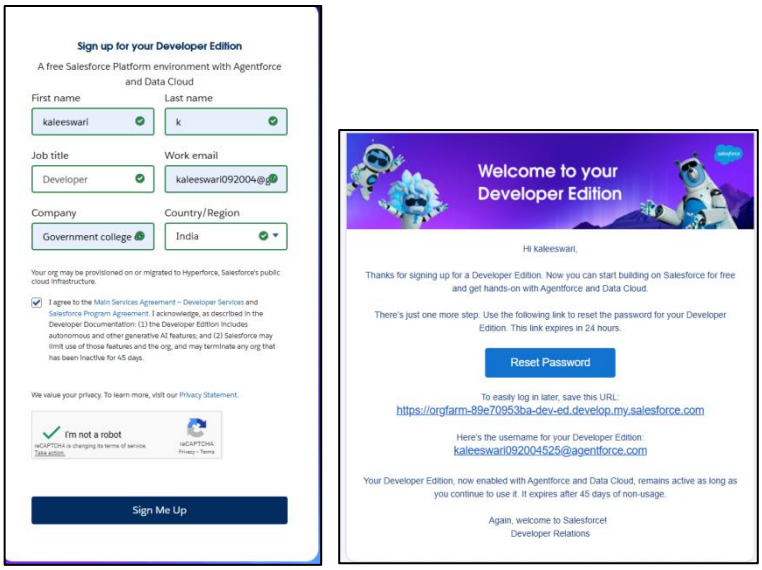
1. Project Milestones & Tasks

The project was executed in multiple milestones, where each milestone focused on a major functional area — from creating Salesforce objects to developing automation and approval workflows.

Milestone No.	Phase / Task	Key Activities Performed	Output
M1	**Salesforce Environment Setup**	Created Salesforce Developer Account and Configured Developer Console	Development environment successfully set up
M2	**Object Creation**	Created 4 Custom Objects: Property, Tenant, Lease, Payment for Tenant using Object Manager	Objects created and linked to database schema
M3	**Custom Tabs & App Setup**	Created Tabs for each object and built a **Lightning App** named "Lease Management" with navigation menus	Fully functional Lease Management Lightning App
M4	**Field Configuration**	Added various **Fields and Data Types** for each object: Property → Name, Address, Type, Sqft Tenant → Email, Phone, Status Lease → Start Date, End Date Payment → Payment Date, Amount, Status	All objects now contain structured data fields ready for input
M5	**Relationship & Validation Setup**	Created Lookup and Master-Detail Relationships between Property, Tenant, Lease, and Payment objects Added Validation Rule for Lease Object (End_Date > Start_Date)	Data relationships and rules successfully enforced

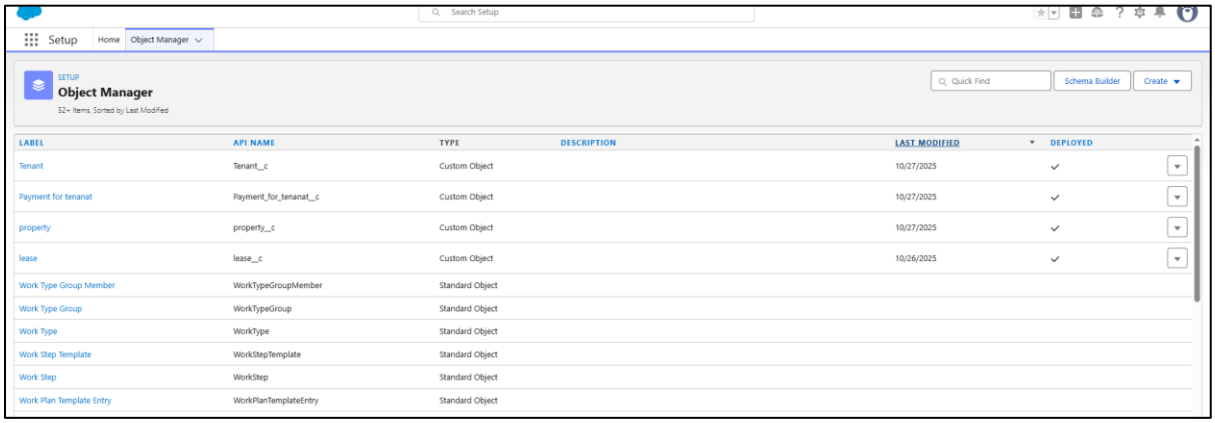
M6	**Email Templates Creation**	Designed **5 Classic Email Templates** for automated communication: Tenant Leaving Leave Approved Leave Rejected Monthly Payment Reminder Payment Success	Email templates saved and available for automation
M7	**Approval Process Implementation**	Created **Approval Process** "Check for Vacant" for Tenant Object to approve or reject tenant leaving requests	Configured automated approval workflow with email alerts for each outcome
M8	**Apex Trigger & Handler Development**	Wrote and tested **Trigger (test)** and **Handler Class (testHandler)** to prevent multiple tenants per property	Data validation logic automated using Apex
M9	**Flow & Scheduled Apex**	Built a **Record-Triggered Flow** for monthly payment confirmation Created and Scheduled **Apex Class (MonthlyEmailScheduler)** for monthly payment reminders	Payment flow automation and scheduled monthly reminder email achieved
M10	**Testing & Debugging**	Verified all functionalities — Email alerts, Approval processes, Triggers, Flows, and Apex Classes	All modules working successfully
M11	**Deployment & Documentation**	Documented all steps and screenshots for the final report	Final working Salesforce-based Lease Management System

Salesforce Environment Setup:



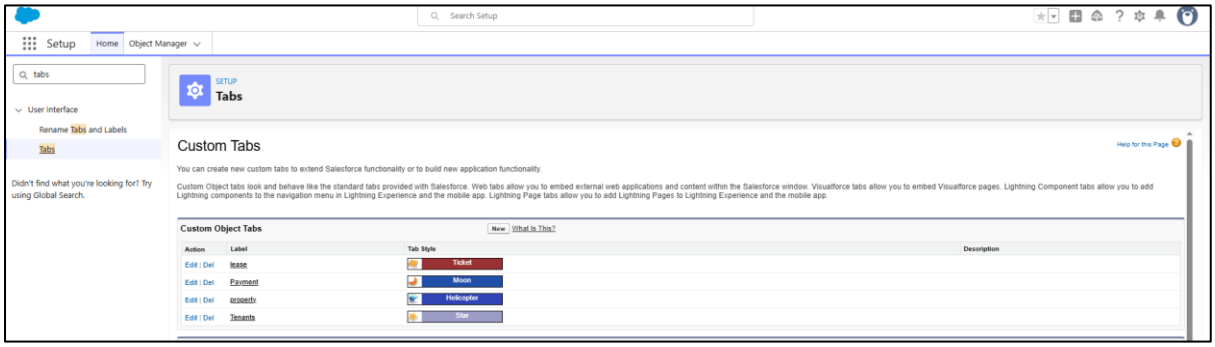
Salesforce Developer Edition environment setup and workspace initialization for the Lease Management System

Object Creation



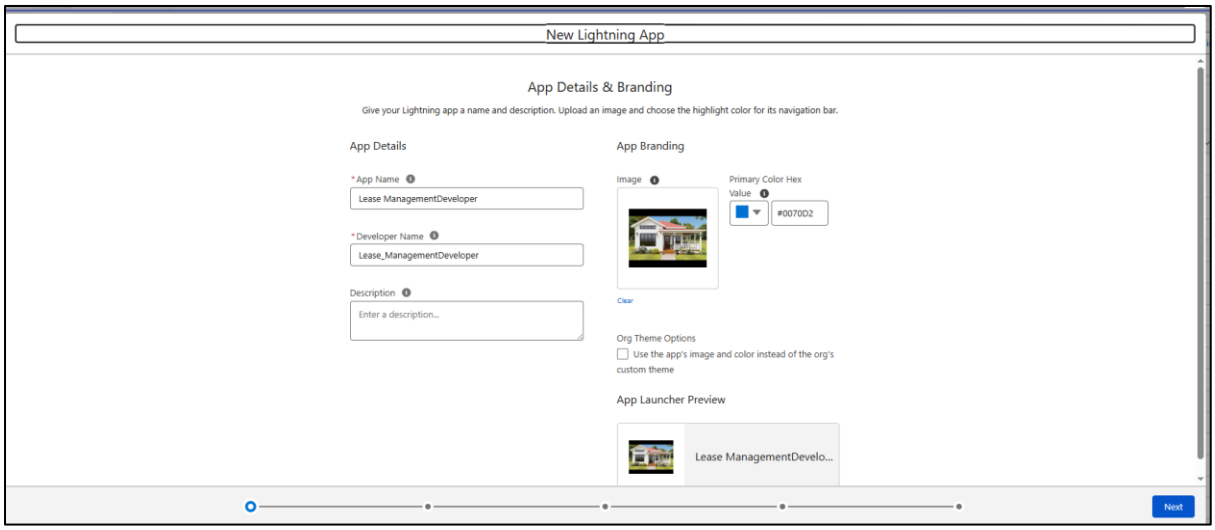
Creation of custom objects (Property, Tenant, Payment, Lease) to structure and store lease-related data

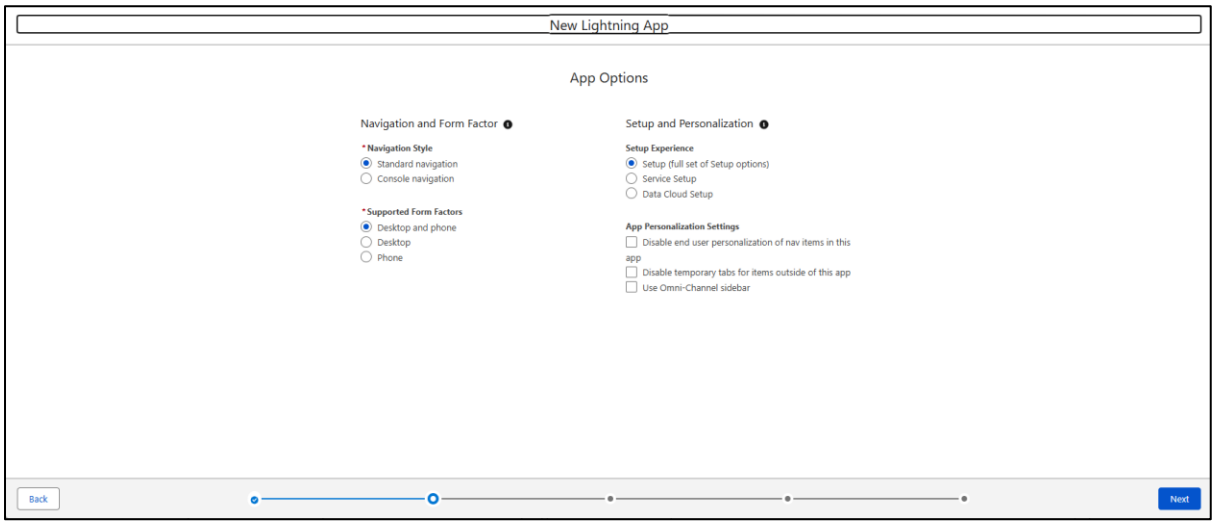
Custom Tabs

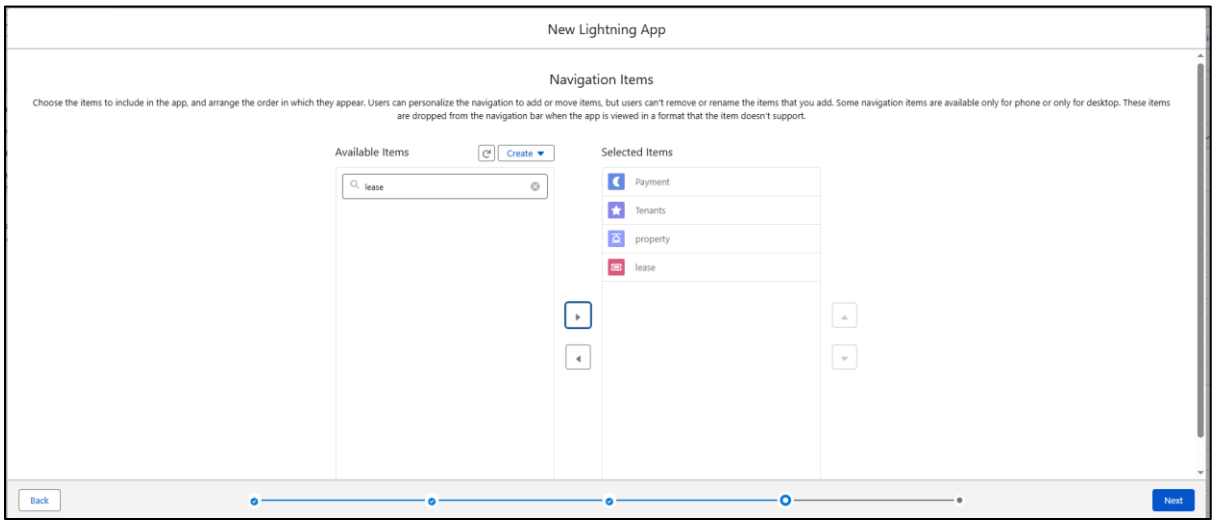


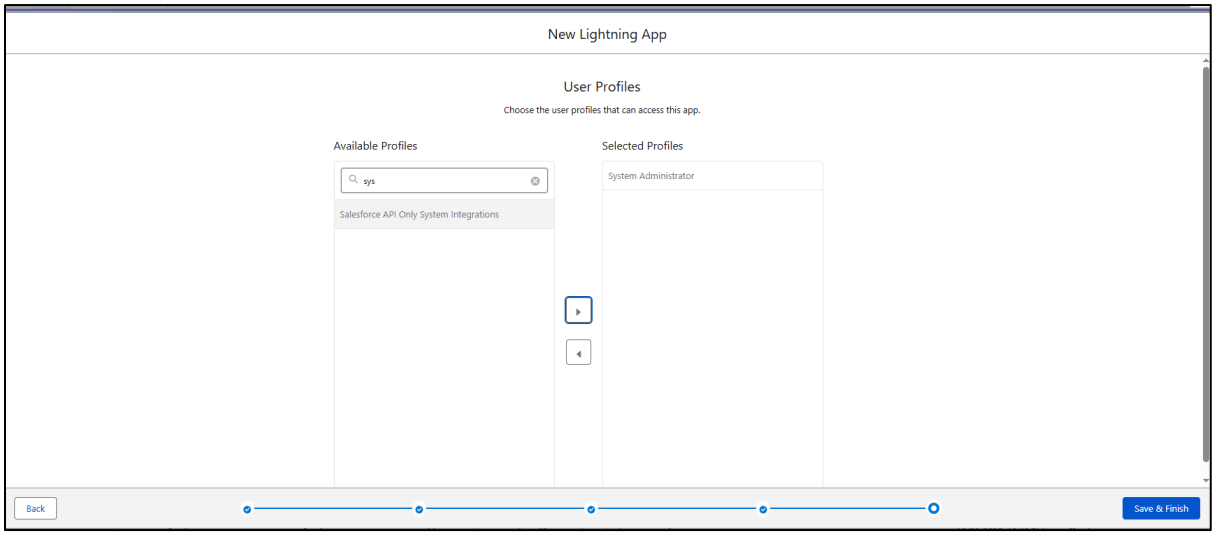
Configuration of custom tabs

App Setup


New Lightning App
App Details & Branding
Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.
App Details
*App Name
Lease ManagementDeveloper
*Developer Name
Lease_ManagementDeveloper
Description
Enter a description...
App Branding
Image
Clear
Primary Color Hex Value
#0070D2
Org Theme Options
Use the app's image and color instead of the org's custom theme
App Launcher Preview
Lease ManagementDevelo...
Next

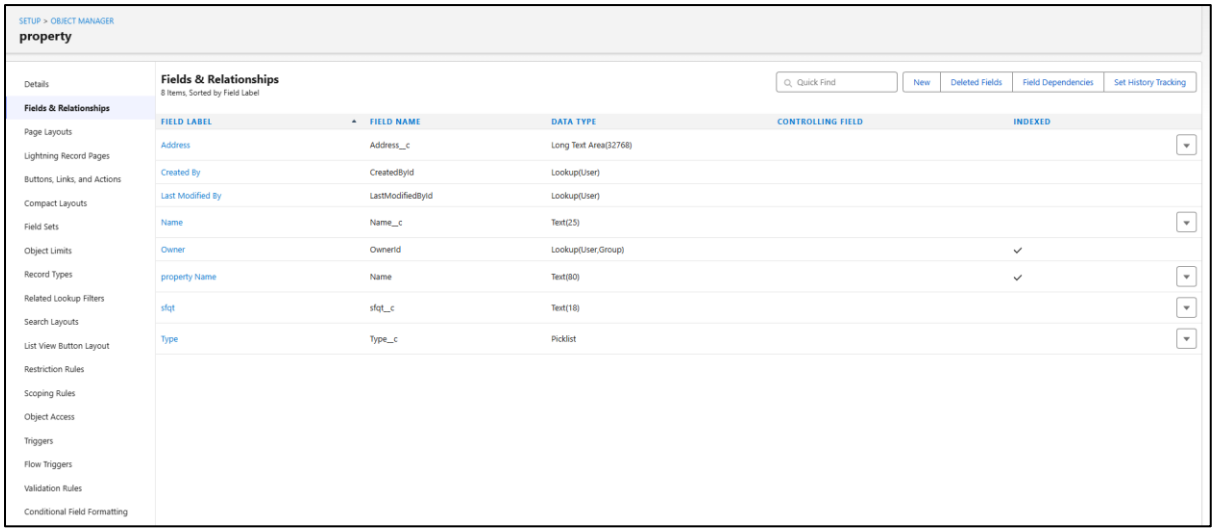

New Lightning App
App Options
Navigation and Form Factor
*Navigation Style
Standard navigation
Console navigation
*Supported Form Factors
Desktop and phone
Desktop
Phone
Setup and Personalization
Setup Experience
Setup (full set of Setup options)
Service Setup
Data Cloud Setup
App Personalization Settings
Disable end user personalization of nav items in this app
Disable temporary tabs for items outside of this app
Use Omni-Channel sidebar
Back
Next


New Lightning App
Navigation Items
Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.
Available Items
Create
lease
Selected Items
Payment
Tenants
property
lease
Back
Next

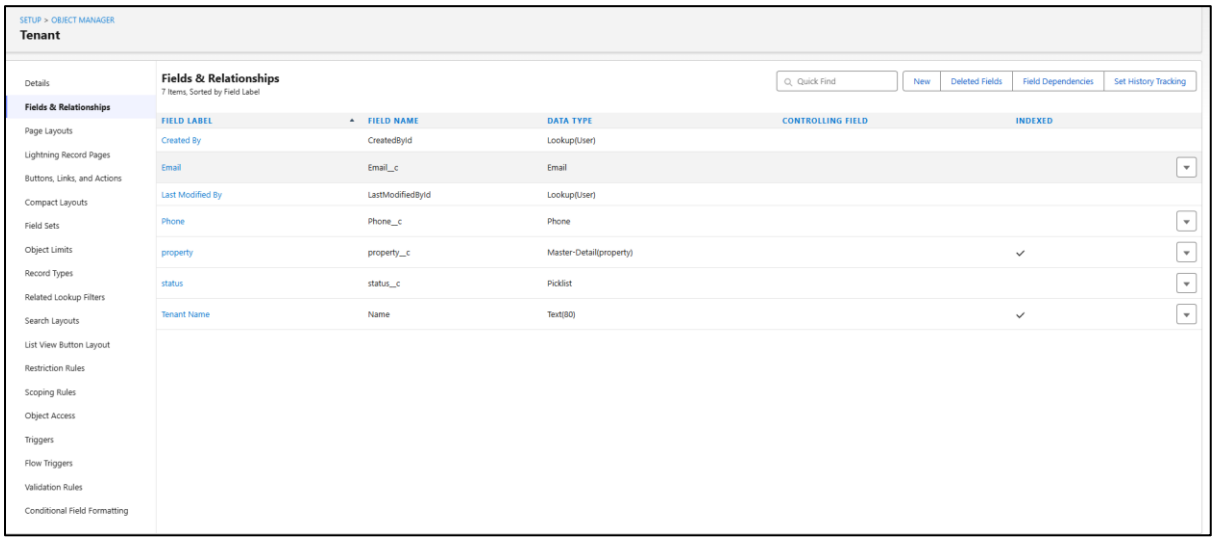


Dedicated Lease Management application for easy navigation within Salesforce

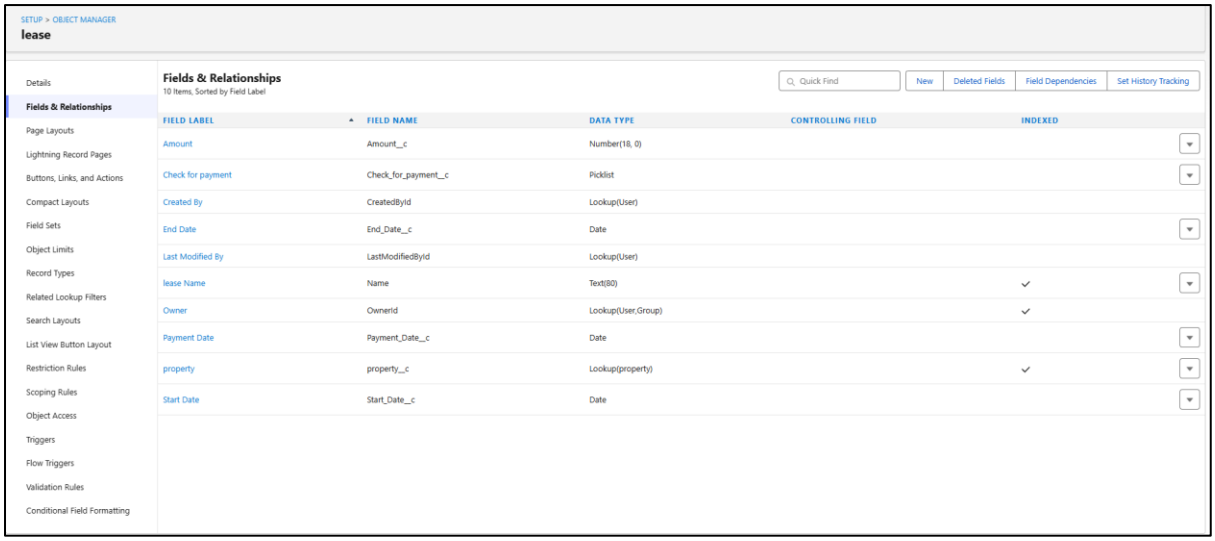
Field Configuration



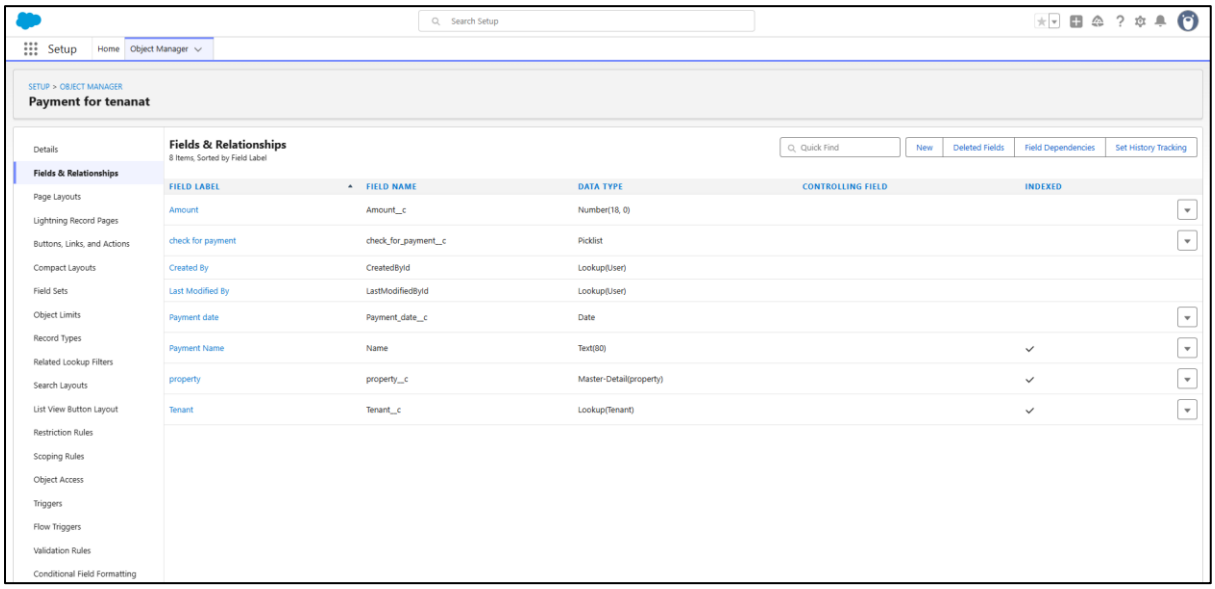
Creation of fields for the property object



Creation of fields for the tenant object

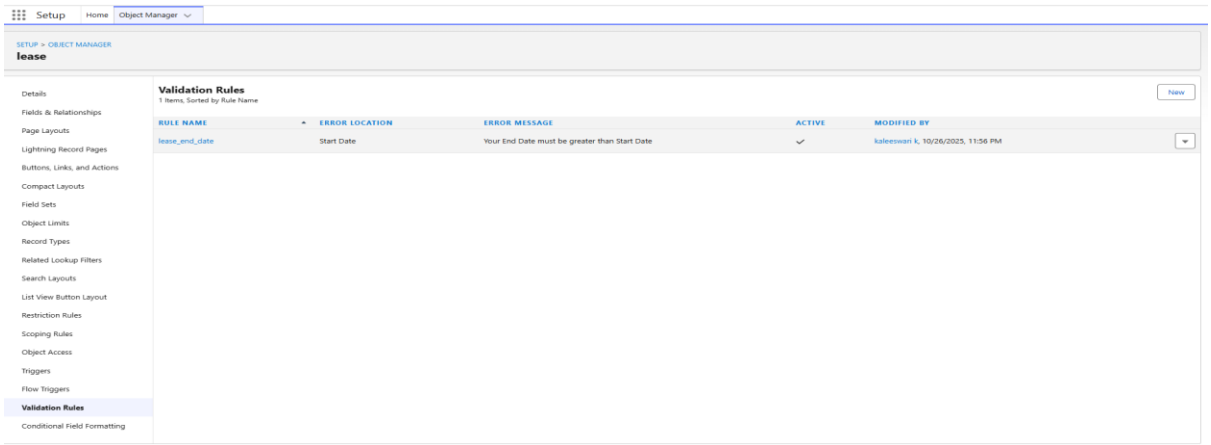


Creation of fields for the lease object

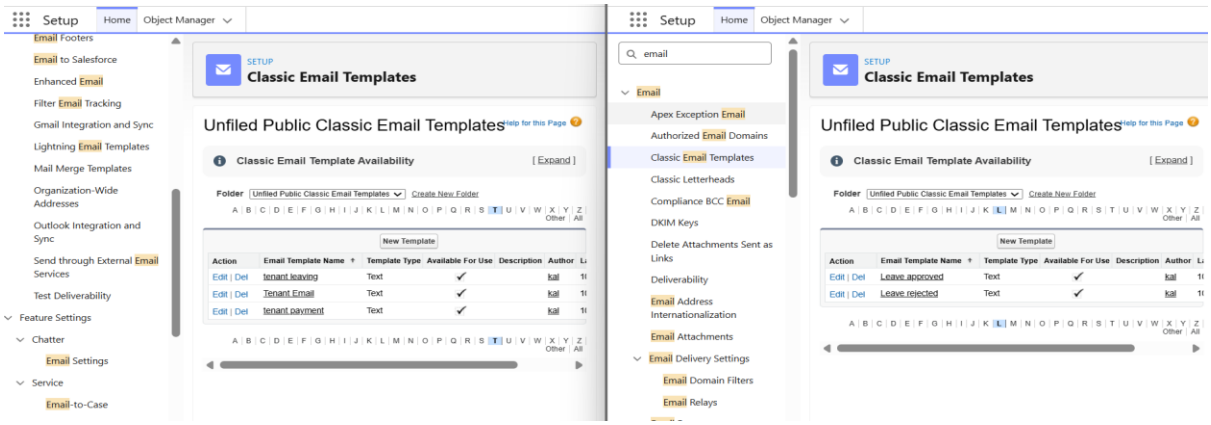


Creation of fields for the Payment for tenant object

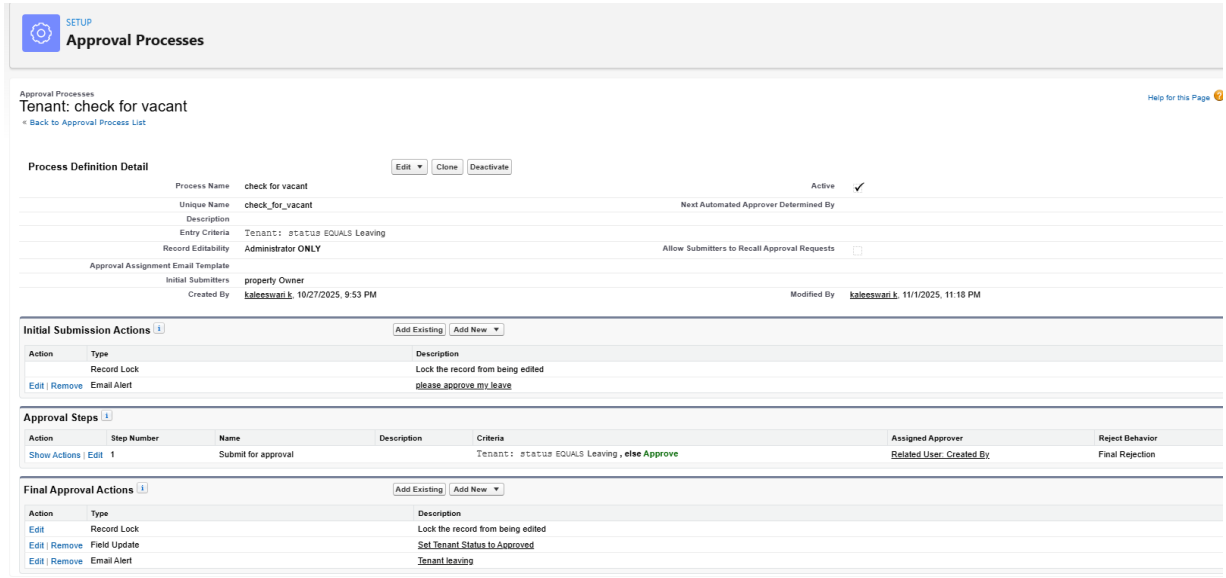
Relationship & Validation Setup



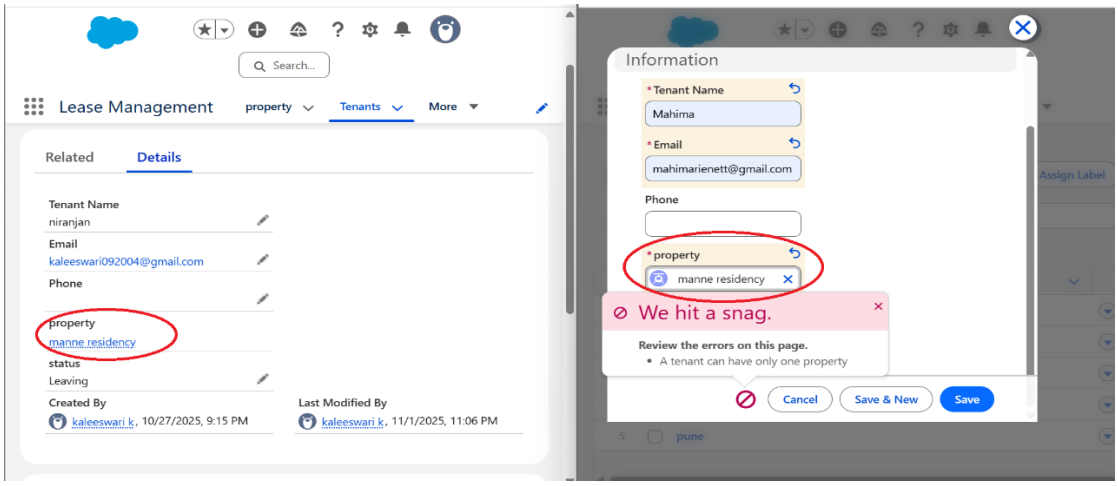
Email Templates Creation



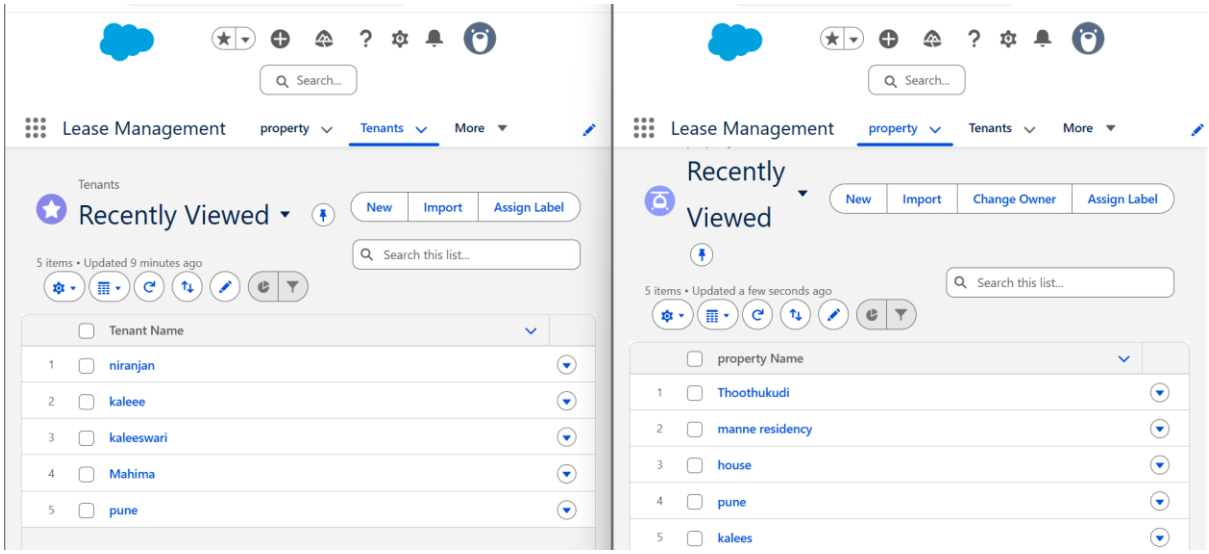
Approval Process Implementation

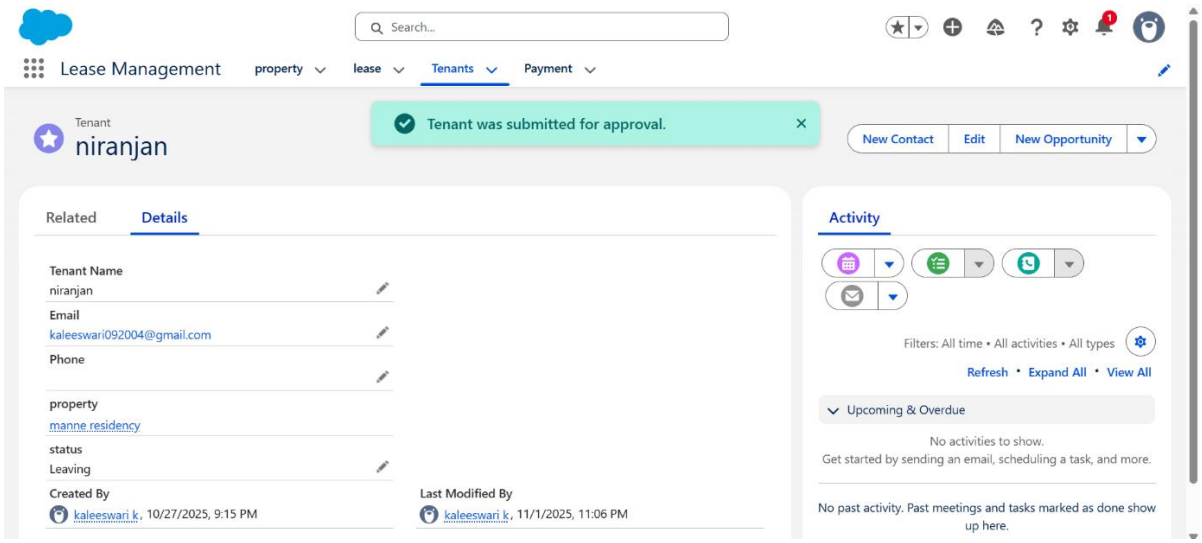


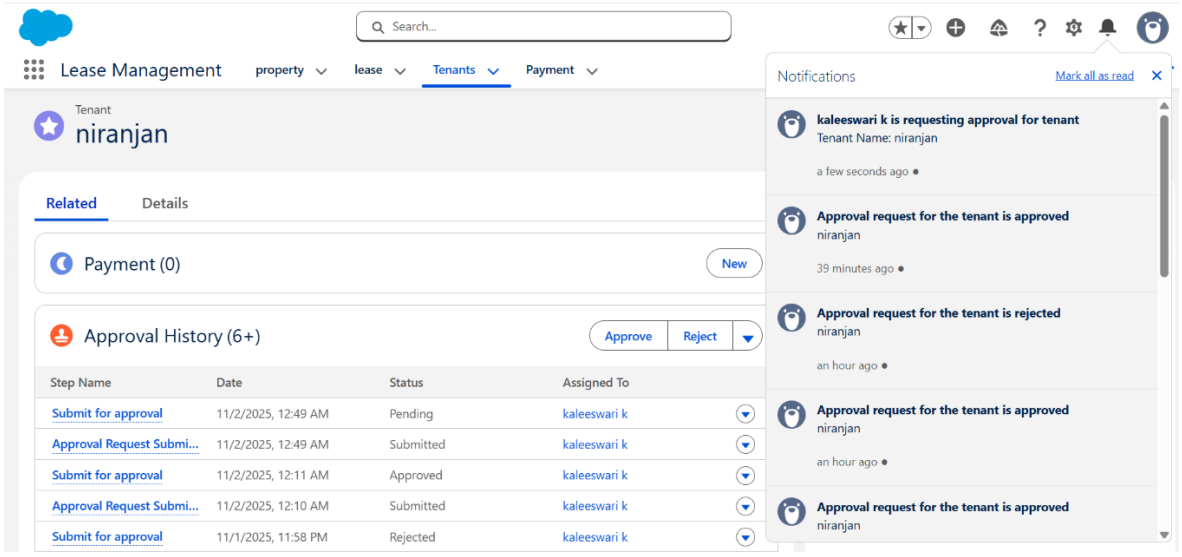
Apex Trigger & Handler Development



Testing & Debugging







2. Sprint Delivery Plan

The development was divided into six agile sprints to ensure incremental progress.

Sprint	Sprint Goal	Deliverables
Sprint 1	Setup & Object Creation	Salesforce org setup, Property/Tenant/Lease/Payment objects created
Sprint 2	Fields & Relationships	Custom fields and lookup/master-detail relationships configured
Sprint 3	App & Tabs Development	Lease Management App and Tabs added
Sprint 4	Automation Setup	Email templates, triggers, and flows implemented
Sprint 5	Approval Workflow	Vacant property approval process created and tested
Sprint 6	Final Testing & Documentation	Error handling, testing, and report preparation completed

3. Project Progress Tracking

To ensure smooth progress, the project followed agile tracking techniques:

- **Daily Logs:** Recorded completion of Salesforce configurations and tests.
- **Sprint Reviews:** Weekly review of created objects, fields, and automation logic.
- **Checkpoints:** Validated Apex, Flow, and Approval process outputs after each sprint.
- **Testing Reports:** Verified that all email alerts, triggers, and validations performed accurately.

Project progress was monitored using:

- **Manual checklists** for Salesforce task completion.
- **Burndown tracking** for pending vs. completed milestones.
- **Screen captures** for every successful output.

4. Team Management Tool for Agile Planning

The project used Trello to manage milestones, tasks, and sprint activities. It helped track which Salesforce features were *Backlog*, *In-Progress*, *Review*, and *Complete*.

Trello Structure Used:

- Board Name: *Lease Management Project*
- Lists: *Backlog*, *In-Progress*, *Review*, and *Complete*
- Cards: Each task (e.g., *Create Lease Object*, *Add Validation Rule*, *Test Apex Trigger*)

6. Final Working of Lease Management Application

Comments **Nira request**

11/2/2025, 12:11 AM

Status **Approved**
Assigned To **kaleeswari k**
Actual Approver **kaleeswari k**
Comments **Nira is approved , byeee**

11/2/2025, 12:10 AM

Status **Submitted**
Assigned To **kaleeswari k**
Actual Approver **kaleeswari k**
Comments **Nira is submitted for approval**

11/1/2025, 11:58 PM

Status **Rejected**
Assigned To **kaleeswari k**
Actual Approver **kaleeswari k**
Comments **Rejected**

11/1/2025, 11:57 PM

Status **Submitted**
Assigned To **kaleeswari k**
Actual Approver **kaleeswari k**
Comments **Leaving Bye**

11/1/2025, 11:36 PM

Status **Approved**
Assigned To **kaleeswari k**
Actual Approver **kaleeswari k**
Comments **approved**

Printable View of Nirajan Tenant