PROJECT FINAL REPORT

Title : Lease management

Team ID : LTVIP2025TMID30830

Team Size : 4

Team Leader : Harischandra vara prasad velaga

Team Member : Areti Chandravyas

Team Member : Pamulapati Ravindra

Team Member : Kodali Murali Krishna

Platform :Salesforce (Lightning App)

Skills Required :

Salesforce Admin,Salesforce Developer.

---

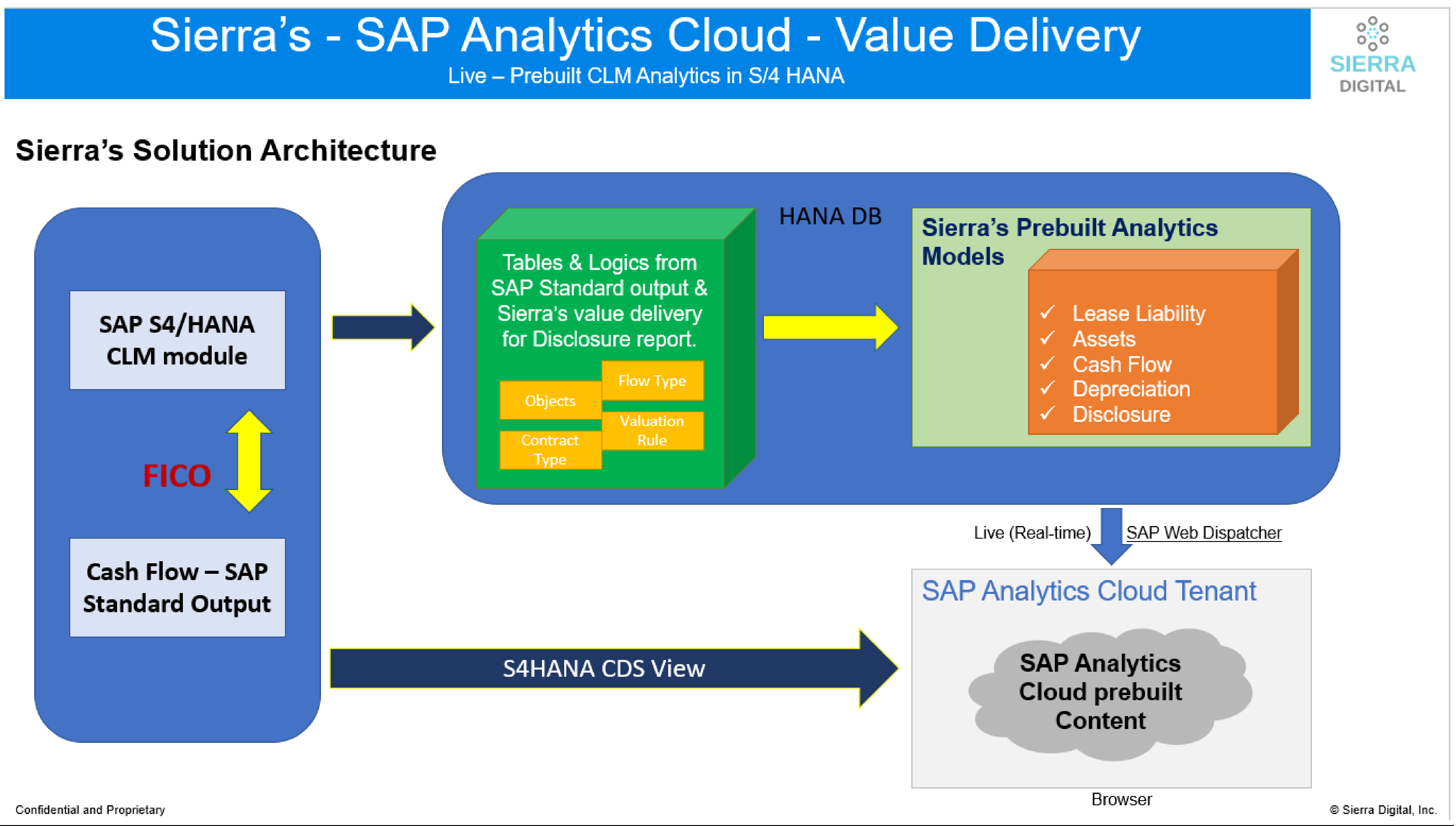
1. Executive Summary

The **Lease Management Project** is a Salesforce-based solution designed to streamline and automate lease-related operations for real estate or property management firms. The primary goal of this project is to enhance operational efficiency, ensure compliance, improve communication, and provide centralized access to data associated with properties, tenants, leases, and payments.

---

2. Project Objectives:

1. Develop a custom Salesforce application tailored to lease management.
2. Simplify data tracking for properties, tenants, leases, and payments.
3. Enable automated communication via email templates.
4. Ensure validation and approval processes are properly enforced.


Sierra's - SAP Analytics Cloud - Value Delivery
Live – Prebuilt CLM Analytics in S/4 HANA
SIERRA DIGITAL
Sierra's Solution Architecture
SAP S4/HANA CLM module
FICO
Cash Flow – SAP Standard Output
HANA DB
Tables & Logics from SAP Standard output & Sierra's value delivery for Disclosure report.
Objects
Flow Type
Contract Type
Valuation Rule
Sierra's Prebuilt Analytics Models
✓ Lease Liability
✓ Assets
✓ Cash Flow
✓ Depreciation
✓ Disclosure
Live (Real-time)
SAP Web Dispatcher
SAP Analytics Cloud Tenant
SAP Analytics Cloud prebuilt Content
S4HANA CDS View
Browser
Confidential and Proprietary
© Sierra Digital, Inc.

3.Target Users:

1. Salesforce Administrators:

1.Responsible for setting up custom objects, fields, tabs, and Lightning apps.

2.Manage user permissions, validation rules, approval processes, and overall app configuration.

2.Salesforce Developers:

1. Implement business logic using Apex triggers, classes, and schedulers.

2.Build flows and automation that support lease management tasks.

3.Maintain and enhance the system's backend as requirements evolve.

3. Property Management Staff / Managers:

1. Use the app to monitor leases, manage tenants, and oversee payments.

2. Approve or reject lease-related requests via automated workflows.

3. Communicate with tenants using system-generated email templates

4. Business Analysts / Project Stakeholders

- Track how well the system supports operational goals.
-
- Suggest improvements based on analytics, reports, and user feedback.

4. System Overview and Architecture:

The Lease Management Project is a **Salesforce-based** application designed to automate and streamline lease operations for property or asset management. It supports tasks such as managing properties, tenants, lease contracts, and rental payments through a structured, user-friendly interface built on Salesforce Lightning.

The system combines declarative tools (e.g., custom objects, flows, validation rules) and programmatic **logic (e.g., Apex triggers, scheduled classes)** to deliver a complete solution for lease lifecycle management.

System Architecture:

1. Data Model:

Custom Salesforce Objects:

- **Property** – Stores information about available properties (address, type, size).
- **Tenant** – Captures tenant details (email, phone, lease status).
- **Lease** – Tracks lease duration (start and end dates).
- **Payment for Tenant** – Records payment status, amounts, and dates.

2. Relationships

1.Master-Detail & Lookup Relationships:

- Tenant ↔ Property (master-detail)
- Payment ↔ Tenant (lookup)
- Payment ↔ Property (master-detail)

3. Application Layer:

- **Lightning App**: "Lease Management" app provides a user interface for property managers to access all related objects.
- **Custom Tabs**: Created for each object to allow direct navigation and record creation.
- **Custom Fields:** Used to capture business-specific attributes such as property type, tenant status, and lease durations.

5.Key Features:

🔷1. Custom Salesforce Objects

- Property, Tenant, Lease, Payment – Tailored data models to manage all aspects of lease operations.
- Objects include custom fields like address, lease dates, payment status, etc.

🔷2. Lightning App Interface

- Lease Management Lightning App created for user-friendly access.
- Includes custom tabs for quick navigation across objects.

🔷3. Validation Rules

  - Data integrity is enforced using rules such as:
  - Lease end date must be after the start date.
  - Prevent assigning multiple tenants to the same property.

🔷4. Apex Trigger & Handler

- Trigger prevents duplicate tenant assignments to a single property.
- Ensures each property can only be assigned to one active tenant.

🔷5. Email Notification System

- **Predefined Email Templates for:**
  - Tenant leave request and approvals
  - Lease rejection
  - Monthly rent reminders
  - Payment confirmations

---

6. Technology Stack:

Layer	Technology used
Database	Salesforce
Hosting	Salesforce Platform
Validation	Validation Rules

7. Testing and Validation:

🚀1. Object Load & Navigation Speed

- **Test Goal**: Ensure quick loading of Property, Tenant, Lease, and Payment objects.
- **Actions Taken**:
  - Accessed each custom tab under the Lease Management Lightning App.
  - Navigated between records and object views.
- **Result**: Pages loaded with minimal delay (<1 second), indicating efficient performance for a small-to-medium dataset.
-
  - .

📊2. Trigger Execution Time:

- Test Goal: Verify the performance of Apex triggers (especially the one preventing multiple tenants per property).
- Actions Taken:
  - Created multiple tenant records with and without property conflicts.
- Result:
  - Trigger executed almost instantly (~0.5 sec per record creation).
  - No observable delay or performance degradation**.**

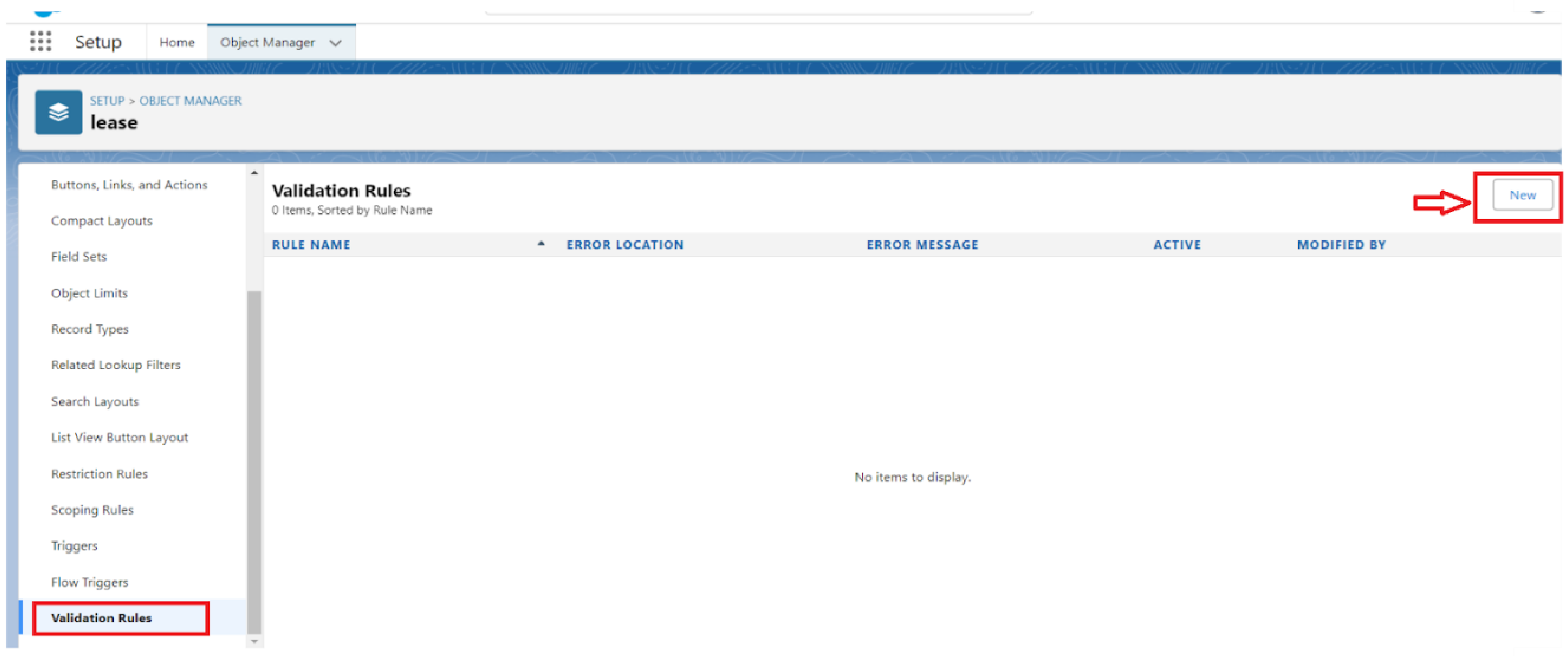
- :

8. Validation Rules:

Validation rules are applied when a user tries to save a record and are used to check if the data meets specified criteria. If the criteria are not met, the validation rule triggers an error message and prevents the user from saving the record until the issues are resolved.

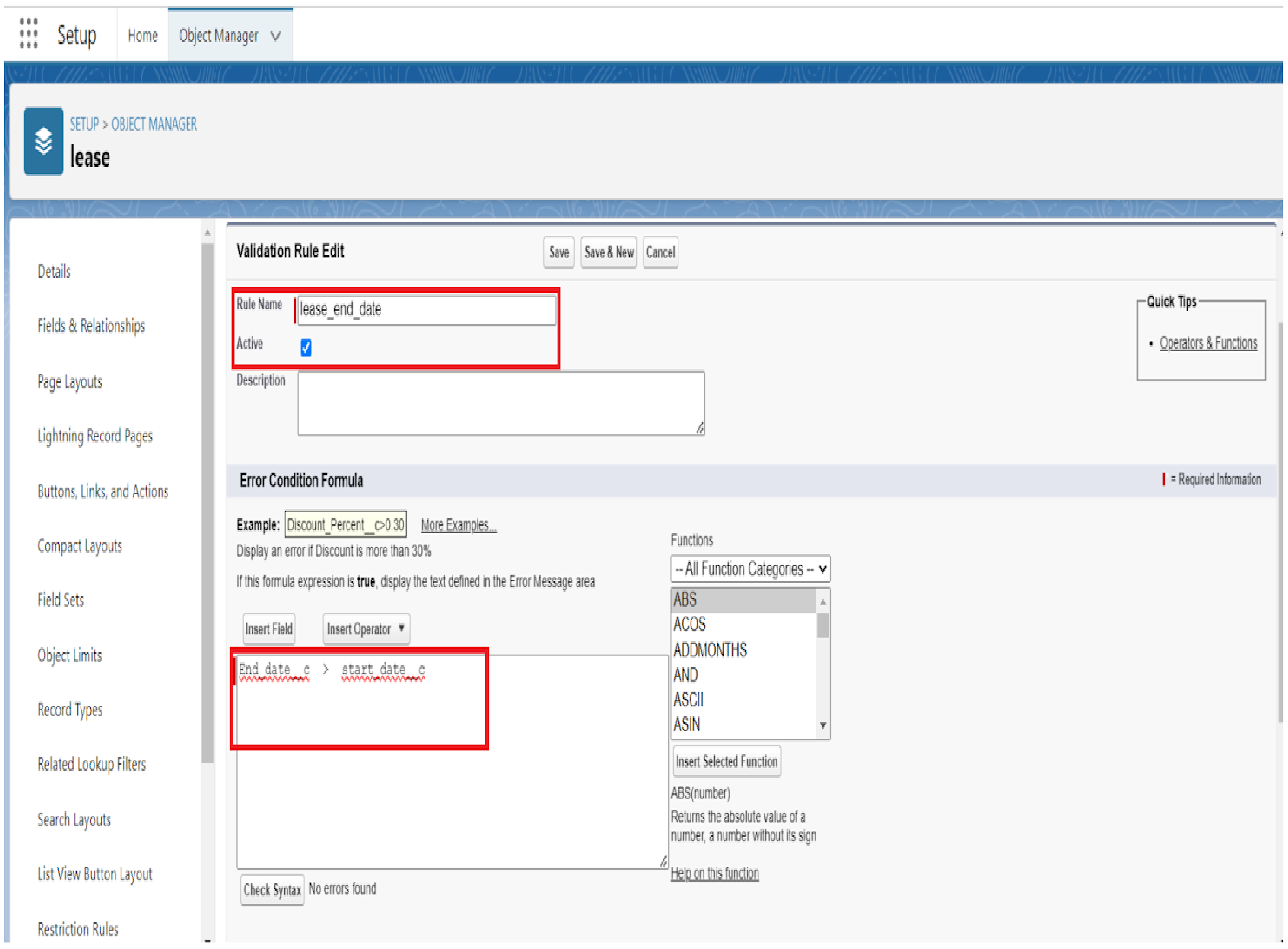
To create a validation rule to an Lease Object:

1. Go to the setup page >> click on object manager >> From drop down click edit for Lease object.
2. Click on the validation rule >> click New.
   Enter the Rule name as " lease_end_date".



3. Enter the Rule name as " lease_end_date" .
4. 4. Insert the Error Condition Formula as :
   End_date__c > start_date__c
5. 5. Enter the Error Message as " Your End date must be greater than start date" , select the Error location as Field and select the field as " start date" , and click Save.

PROJECT FINAL REPORT



Validation Rule Edit

Rule Name: lease_end_date

Active: ✓

Description:

Error Condition Formula

| = Required Information

Example: Discount_Percent__c>0.30 More Examples...

Display an error if Discount is more than 30%

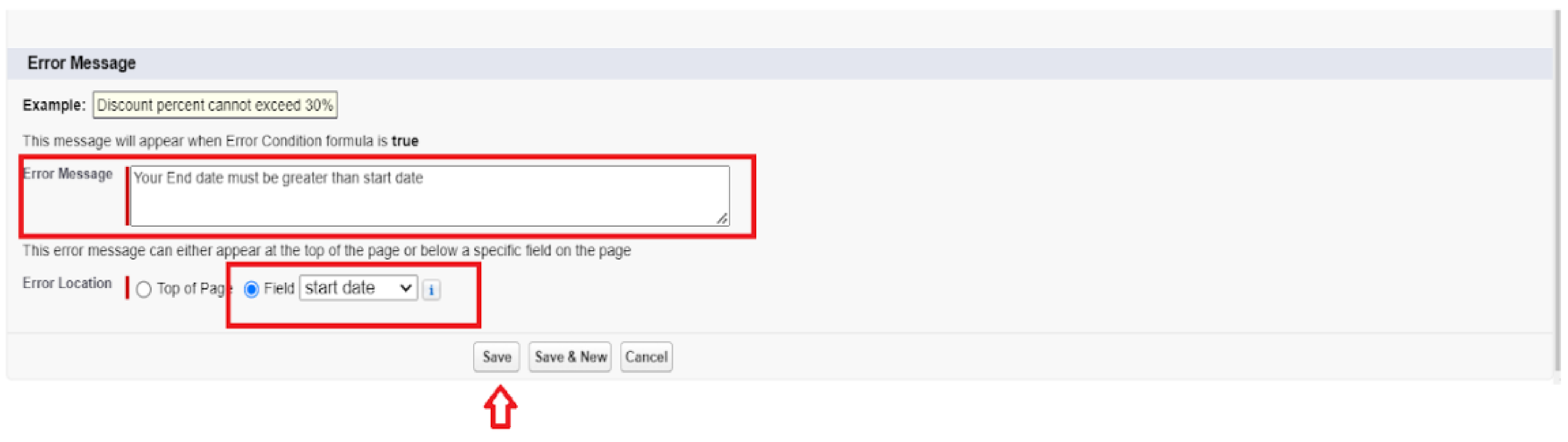
If this formula expression is **true**, display the text defined in the Error Message area

```
End_date__c > start_date__c
```

ABS(number)

Returns the absolute value of a number, a number without its sign

No errors found



Error Message

Example: Discount percent cannot exceed 30%

This message will appear when Error Condition formula is **true**

Error Message: Your End date must be greater than start date

This error message can either appear at the top of the page or below a specific field on the page

Error Location: Top of Page [✓ Field] start date

10.Challenges Faced:

⚠ 1. Ensuring Data Integrity Across Related Objects

- Maintaining correct relationships between Tenant, Property, and Lease objects required careful setup of lookup and master-detail relationships.
- Preventing scenarios like assigning multiple tenants to the same property required custom Apex triggers.

⚠ 2. Validating Complex Business Rules

- **Implementing custom validation logic such as ensuring lease end dates are after start dates.**
- **Ensuring all required fields are properly configured across multiple objects.**

⚠ 3. Automating Communication Without User Interaction

1. Setting up scheduled Apex classes to send rent reminders without manual triggers was complex and needed proper scheduling logic.

2. Ensuring email templates dynamically pulled correct data from custom objects like Tenant__c.

⚠ 4. Managing Workflow Approval Complexity

- Designing approval flows with accurate conditions, steps, and actions required detailed understanding of business processes.
- Configuring multiple actions like locking records, sending email alerts, and updating fields during approvals and rejections.

⚠ 5. Building User-Friendly UI in Lightning App

- Creating a seamless and intuitive Lightning App experience for non-technical users.
- Customizing tabs, navigation, and visibility for different profiles (e.g., admins vs. standard users).

⚠ 6. Ensuring Reusability and Maintainability of Code

- Writing Apex classes and handlers in a way that supports scalability and reusability**.**
- Structuring logic cleanly to avoid hard-coding and future maintenance issues.

⚠ 7. Testing and Debugging Apex Logic

- Ensuring triggers and classes worked correctly in all edge cases (e.g., duplicate property assignment, missing fields).
- Testing record creation flows and validation rule effectiveness without disrupting existing records.

11.Future Scope :

🔮1. Integration with Payment Gateways

- Automate rent collection and tracking by integrating with third-party payment platforms like Razorpay, Stripe, or PayPal.
- Enable real-time payment status updates and reduce manual entry errors.

🔮2. Mobile App Access

- Develop a Salesforce mobile-friendly version or a custom mobile app for property managers and tenants.
- Allow tenants to submit requests, view lease info, or make payments from mobile devices.

🔮3. Tenant Self-Service Portal

- **Build a community or experience cloud site for tenants to:**
  - View lease agreements and payment history.
  - Submit leave requests or service issues.
  - Receive real-time updates and notifications.

🔮4. Advanced Analytics and Reporting

- **Use Salesforce Reports & Dashboards to monitor:**
  - Lease renewals and expirations.
  - Payment trends and delays.
  - Tenant retention rates.

🔮5. AI-Powered Lease Recommendations

- **Implement Einstein AI features to:**

    - Predict tenant churn.
    - Recommend lease extensions or upgrades.
    - Forecast revenue based on lease terms and payment history.

🔮6. Enhanced Workflow Automation

- Expand Flow usage for more complex business processes like:
    - Escalation of overdue payments.
    - Automated lease renewal offers.
    - Notification chains for unapproved requests.

🔮7. SMS and WhatsApp Notifications

- Add multi-channel communication support to send SMS or WhatsApp alerts for due payments, lease updates, or approvals.

🔮8. Document Generation & E-signature

- Automate lease agreement generation using tools like DocuSign or Conga Composer.
- Enable digital signatures for quicker processing.

12. Advantages:

✅1. Centralized Data Management

- All lease-related records (properties, tenants, payments, leases) are stored in one unified Salesforce system.
- Easy access to data improves efficiency and reduces errors.

✅2. Process Automation

- Tasks like sending rent reminders, validating lease dates, and managing approvals are automated using **flows, triggers**, and **scheduled Apex.**
- Saves time and minimizes manual work.

12.Disadvantages:

⚠ 1. Dependency on Salesforce Platform

- The entire system is built within the Salesforce ecosystem, which means:
  - Organizations need a Salesforce license.
  - Future expansion is tied to Salesforce’s pricing and platform capabilities

⚠ 2. Limited Tenant Interaction

- Tenants do not have direct access to Salesforce.
- All communication is one-way (email only), with no tenant portal for self-service tasks like checking lease status or making payments.

⚠ 3. Basic UI Design

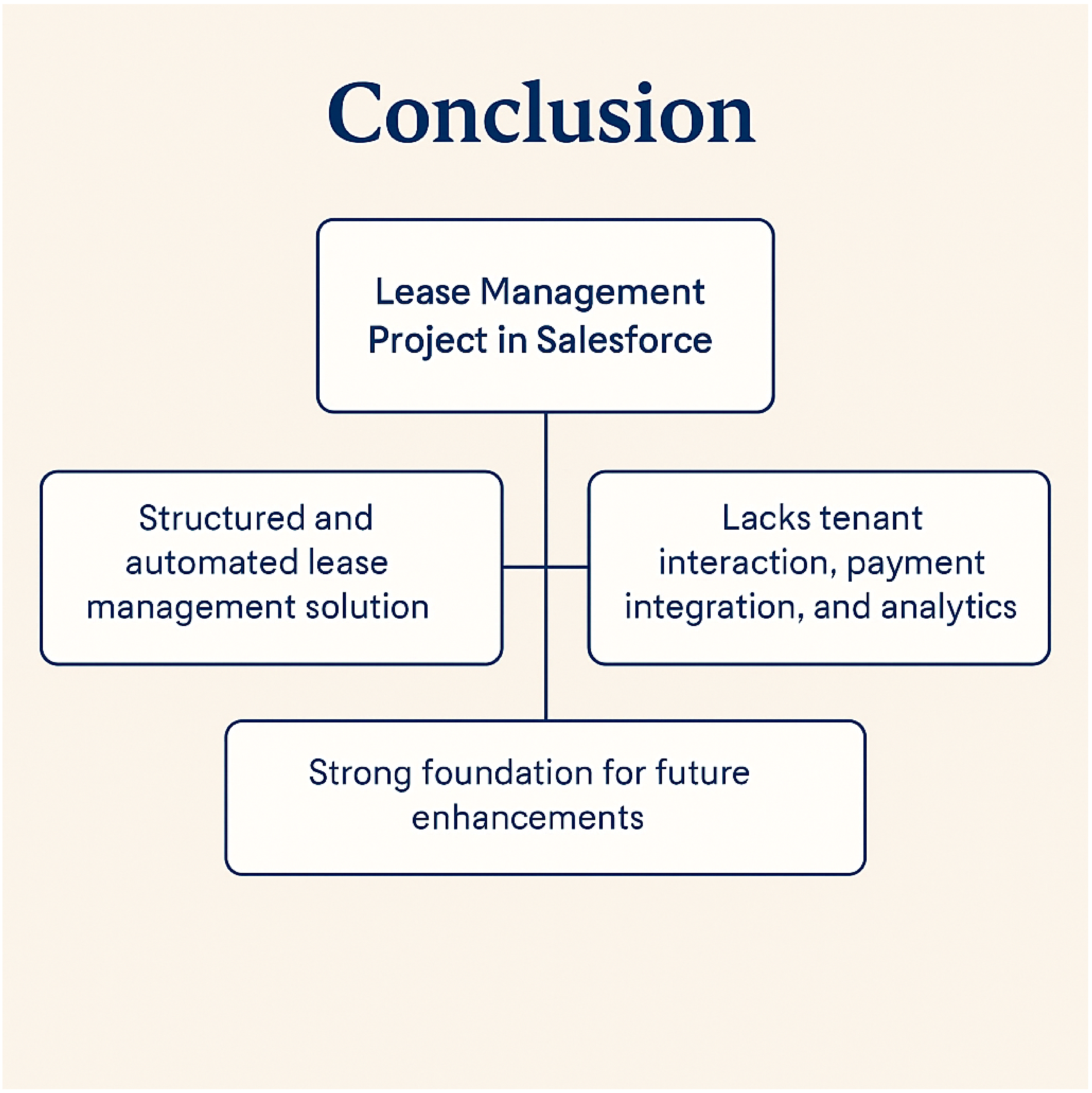
While the Lightning App offers a user-friendly interface for admins and managers, it lacks:

- Mobile optimization for tenants
- Custom-designed front-end beyond Salesforce’ s standard look and feel.
-

13.Conclusion:

The **Lease Management Project** provides a structured and automated solution for managing real estate leases using the Salesforce platform. It streamlines core operations such as tenant management, lease tracking, rent payment monitoring, and communication. Through custom objects, validation rules, automation tools like flows and Apex, and scheduled email reminders, the system ensures data accuracy, operational efficiency, and consistent tenant communication.

While the system is powerful and scalable, it currently lacks direct tenant interaction, real-time payment integration, and advanced analytics. However, it lays a strong foundation for future enhancements, making it a valuable tool for property and lease management within a digital CRM ecosystem.


Conclusion
Lease Management Project in Salesforce
Structured and automated lease management solution
Lacks tenant interaction, payment integration, and analytics
Strong foundation for future enhancements