

# **FinSmart CRM – Loan & Customer Relationship Management System**

**Industry:** Banking & Financial Services

**Target Users:** Loan Officers, Customers, Risk Analysts, and Bank Managers

## **Overview:**

This project focuses on building a Salesforce-based **FinSmart CRM** solution to improve the bank's loan management process. The system will automate loan application intake, verification, and approval to reduce delays and manual errors. Customers will get a self-service portal to apply for loans, track their status, and receive automatic reminders for repayment schedules and overdue amounts. Integration with credit bureaus will allow instant credit score checks and personalized loan offers. Managers will benefit from real-time dashboards and reports that provide insights into loan portfolio health, repayment performance, risks, and revenue growth. Overall, the project aims to increase efficiency, reduce defaults, improve customer satisfaction, and support smarter decision-making.

## **Problem Statement:**

A leading bank receives thousands of loan applications every month across different channels such as branch offices, online forms, and partner agencies. However, the loan approval and repayment management process faces multiple challenges:

- Loan application intake and approvals are **manual and time-consuming**, leading to delays.
- Repayment schedules are **not tracked efficiently**, causing high default risks.
- Customers do not receive **personalized offers** based on their credit history and past transactions.
- Managers lack **real-time visibility** into portfolio risk, loan performance, and defaulter trends.

To address these issues, the bank wants to implement a **Salesforce-based FinSmart CRM** that will include following **Objective**:

- Automate **loan application intake, verification, and approval workflows**.
- Provide **customers with a portal** to apply for loans and track their status.
- Send **automated reminders** for repayment schedules and overdue accounts.
- Integrate with **credit bureau APIs** for credit score checks.
- Deliver **dashboards and reports** on loan portfolio health, repayment performance, and revenue growth.



## Phase 1: Problem Understanding & Industry Analysis

### 1. Requirement Gathering

The banking institution currently faces multiple operational challenges:

- Manual loan application intake through branches and online forms results in **delays in approval processing**.
- Repayment schedules are maintained in **separate systems**, causing **missed follow-ups and defaults**.
- Customers do not receive **personalized loan offers**, limiting engagement.
- Managers lack **real-time visibility** into loan portfolio performance and risks.

The key requirements identified are:

- Automate **loan intake and approval workflows**.
- Provide **customer self-service portals** for loan tracking and document uploads.
- Automate **repayment reminders and overdue alerts**.
- Integrate with **credit bureau APIs** to validate credit scores.
- Create **dashboards** for loan performance, defaults, and revenue tracking.

## 2. Stakeholder Analysis

- **Customers (Borrowers):** Need a transparent and fast loan application process, repayment reminders, and personalized offers.
- **Loan Officers:** Require tools to manage loan applications, verify documents, and handle escalations efficiently.
- **Risk Analysts:** Need insights into credit history, defaulter trends, and portfolio risks.
- **Managers:** Require real-time dashboards on loan approvals, repayments, defaults, and revenue growth.
- **System Admin/Developers:** Need clear data models, automation, and integrations to ensure smooth operations.

## 3. Business Process Mapping

### Current Process:

Loan Application (manual form) → Document Verification (offline) → Credit Check (delayed) → Manager Approval → Disbursement → Repayment Tracking (manual).

### Proposed Salesforce-Enabled Process:

Loan Application (online portal) → Automated Document Verification → Real-time Credit Bureau Check → Automated Approval Workflow → Loan Disbursement → Repayment Reminders (Flows/Email/SMS) → Portfolio Dashboards.

## 4. Industry-specific Use Case Analysis

In the **Banking & Financial Services** industry, digital loan management systems are standard for improving efficiency. Competitors use advanced CRM solutions for:

- **Faster loan approvals** through automation.
- **Lower default rates** with repayment reminders.
- **Higher customer satisfaction** via self-service portals.
- **Cross-selling opportunities** using personalized offers.

FinSmart CRM will bring these industry best practices into the bank's operations.

## 5. AppExchange Exploration

Relevant Salesforce AppExchange apps to consider:

- **Credit Bureau Integration Apps** (for credit score checks).
- **Payment Gateway Connectors** (for EMI collection).

- **Document Management Apps** (for KYC storage).
- **Banking Analytics Dashboards** (pre-built loan performance report)

# Phase 2: Org Setup & Configuration

## 1. Company Profile Setup

What I did: I configured the Company Information in Salesforce with the name 'FinSmart Bank', locale as 'English (India)', time zone as '(GMT+05:30) Kolkata', and default currency as 'INR'.

Reason: This ensures all records and transactions align with the bank's working region, and financial values are consistently reported in Indian Rupees.

The screenshot shows the Salesforce Setup interface with the following details:

- Left Sidebar:** Shows a search bar with "Q\_ company" and a list of categories under "Company Settings": Business Hours, Calendar Settings, Public Calendars and Resources, **Company Information** (which is selected), Data Protection and Privacy, Fiscal Year, Holidays, Language Settings, and My Domain.
- Top Bar:** Includes "SETUP", "Home", "Object Manager", and various navigation icons.
- Page Header:** "Company Information" with a "General Information" section.
- General Information:** Organization Name: FinSmart Bank, Primary Contact: OrgFarm EPIC, Division: (empty), Phone: (empty), Fax: (empty).
- Address:** Country: India, Street: (empty), City: (empty), State/Province: --None--, Zip/Postal Code: (empty).
- Locale Settings:** Default Locale: English (India), Default Language: English, Default Time Zone: (GMT+05:30) India Standard Time (Asia/Kolkata).
- Currency Settings:** (Not visible in the screenshot)

## 2. Fiscal Year

What I did: I set up a Standard Fiscal Year starting in April.

Reason: Banks in India follow the April–March financial cycle. This setup ensures reports and analytics align with regulatory and compliance needs.

**Fiscal Year Information**

Your organization can change the fiscal year start month, and specify whether the fiscal year name is set to the starting or ending year. For example, if your fiscal year starts in April 2025 and ends in March 2026, your Fiscal Year setting can be either 2025 or 2026.

**Change Fiscal Year Period**

Name:	FinSmart Bank	<input type="button" value="Save"/>	<input type="button" value="Cancel"/>
Fiscal Year Start Month:	April	<input type="button" value=""/>	
Fiscal Year is Based On:	<input type="radio"/> The ending month <input checked="" type="radio"/> The starting month	<input type="button" value=""/>	

### 3. Business Hours & Holidays

What I did: I created business hours named 'FinSmart Business Hours' with working hours from 09:00AM to 6:00PM. I also added holidays to reflect official bank holidays.

Reason: Defining business hours ensures that service level agreements, workflows, and escalations respect actual working time. Adding holidays prevents automated tasks from running when the bank is closed.

**Business Hours Edit**

**Step 1. Business Hours Name**

Business Hours Name:  \* = Required Information

Active:

Use these business hours as the default:

**Step 2. Time Zone**

Time Zone:

**Step 3. Business Hours**

Day	From	To	24 hours
Sunday	9:00AM	12:00PM	<input type="checkbox"/>
Monday	9:00 AM	6:00 PM	<input type="checkbox"/>
Tuesday	9:00 AM	6:00 PM	<input type="checkbox"/>
Wednesday	9:00 AM	6:00 PM	<input type="checkbox"/>
Thursday	9:00 AM	6:00 PM	<input type="checkbox"/>
Friday	9:00 AM	6:00 PM	<input type="checkbox"/>
Saturday	HH:MM	HH:MM	<input type="checkbox"/>

The screenshot shows the Salesforce Setup interface with the 'Holidays' object selected. The left sidebar has 'Company Settings' expanded, with 'Holidays' highlighted. A search bar at the top left contains 'Holidays'. The main content area title is 'SETUP Holidays'. Below it, a section titled 'Holidays' describes them as dates and times when business hours are suspended. A table lists several holidays with their names, descriptions, and dates/times. A note indicates 'Holidays ~ Salesforce - Developer Edition' is visible. Below the table is a section titled 'Elapsed Holidays' which says 'No records to display'.

## 4. My Domain

What I did: I registered a unique domain name (e.g., finsmart-bank) and deployed it to users.

Reason: A My Domain is required for Lightning components, LWCs, and connected apps. It also improves branding and login security for the organization.

The screenshot shows the Salesforce Setup interface with 'My Domain' selected in the 'Company Settings' section of the left sidebar. A search bar at the top left contains 'my'. The main content area title is 'SETUP My Domain'. Below it, a section titled 'My Domain Settings' describes how it showcases the company's brand and keeps data secure. A note states that domains include the company-specific My Domain name. The 'My Domain Details' section shows the current URL as 'finsmart-crm-dev-ed.develop.my.salesforce.com' with partitioned enhanced domains, and the my domain name as 'finsmart-crm-dev-ed'. The 'Domain Suffix' is listed as 'Standard (\*.my.salesforce.com)'. The 'Routing and Policies' section includes a note about Salesforce Edge Network and a 'Login Policy' section with a checkbox for preventing login from specific URLs.

## 5. Org-Wide Defaults (OWD)

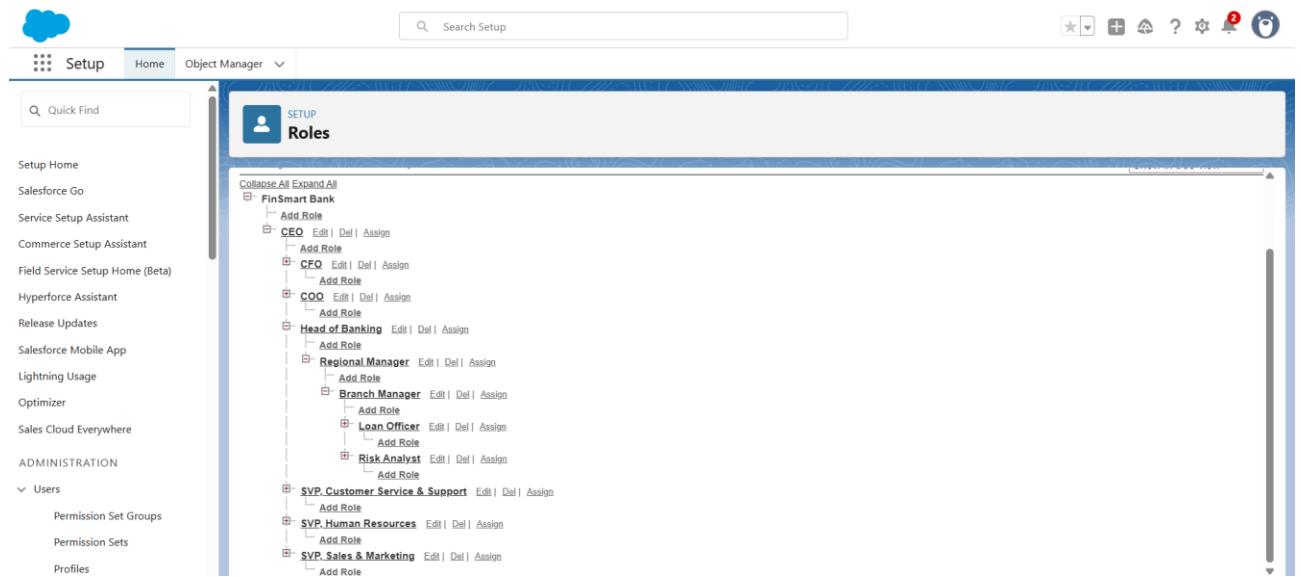
What I did: I planned OWD settings for Customer\_\_c, Loan\_Application\_\_c, and Repayment\_\_c as Private.

Reason: The actual configuration will be done in later stages of the project.

## 6. Role Hierarchy

What I did: I created a hierarchy: Head of Banking → Regional Manager → Branch Manager → (Loan Officer, Risk Analyst).

Reason: This role structure ensures proper data visibility flows upwards, allowing managers to monitor their teams while restricting unnecessary access at lower levels.



The screenshot shows the Salesforce Setup Roles page. The sidebar on the left lists various setup categories like Setup Home, Service Setup Assistant, Commerce Setup Assistant, etc. The main content area has a search bar and a title 'SETUP Roles'. A tree view displays the role hierarchy:

- FinSmart Bank
  - Add Role
  - CEO
    - Edit | Del | Assign
    - Add Role
  - CFO
    - Edit | Del | Assign
    - Add Role
  - COO
    - Edit | Del | Assign
    - Add Role
  - Head of Banking
    - Edit | Del | Assign
    - Add Role
  - Regional Manager
    - Edit | Del | Assign
    - Add Role
  - Branch Manager
    - Edit | Del | Assign
    - Add Role
  - Loan Officer
    - Edit | Del | Assign
    - Add Role
  - Risk Analyst
    - Edit | Del | Assign
    - Add Role
  - SVP\_Customer\_Service & Support
    - Edit | Del | Assign
    - Add Role
  - SVP\_Human\_Resources
    - Edit | Del | Assign
    - Add Role
  - SVP\_Sales & Marketing
    - Edit | Del | Assign
    - Add Role

## 7. Profiles & Permission Sets

What I did: I cloned the Standard User profile to create three profiles:

- Loan\_Officer\_Profile – for loan officers who handle customer loan applications.
- Risk\_Analyst\_Profile – for risk analysts who verify documents and assess loan risk.
- Manager\_Profile – for managers who oversee customers, loans, repayments, and documents.

Additionally, I planned to create a Permission Set for extra access requirements. This will be done in later phases.

Reason: Profiles control object permissions. Custom profiles were required to match responsibilities, while Permission Sets provide flexible access extensions without changing the core profiles.

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Cloud icon, search bar ("Search Setup"), and various setup icons.
- Breadcrumbs:** Setup > Object Manager > Profiles
- Search Bar:** "profiles"
- Left Sidebar:** "Users" section with "Profiles" selected.
- Message:** "Didn't find what you're looking for? Try using Global Search."
- Profile Detail:**

Name	Loan_Officer_Profile		
User License	Salesforce	Custom Profile	✓
Description			
Created By	Shrinidhi Shrinidhi, 9/21/2025, 9:13 AM	Modified By	Shrinidhi Shrinidhi, 9/21/2025, 9:13 AM
- Page Layouts:**

Object	Layout	Assignment
Global	Global Layout [ View Assignment ]	Location Group Assignment Layout [ View Assignment ]
Email Application	Not Assigned [ View Assignment ]	Macro Layout [ View Assignment ]
Home Page Layout	Home Page Default [ View Assignment ]	Object Milestone Layout [ View Assignment ]
Account	Account Layout [ View Assignment ]	Operating Hours Operating Hours Layout [ View Assignment ]
Alternative Payment Method	Alternative Payment Method Layout [ View Assignment ]	Opportunity Opportunity Layout [ View Assignment ]
Appointment Invitation	Appointment Invitation Layout [ View Assignment ]	Opportunity Product Opportunity Product Layout [ View Assignment ]
Asset	Asset Layout [ View Assignment ]	Order Order Layout [ View Assignment ]

The screenshot shows the Salesforce Setup interface with the following details:

- Header:** Cloud icon, search bar ("Search Setup"), and various setup icons.
- Breadcrumbs:** Setup > Object Manager > Profiles
- Search Bar:** "profiles"
- Left Sidebar:** "Users" section with "Profiles" selected.
- Message:** "Didn't find what you're looking for? Try using Global Search."
- Profile Detail:**

Name	Risk_Analyst_Profile		
User License	Salesforce	Custom Profile	✓
Description			
Created By	Shrinidhi Shrinidhi, 9/21/2025, 9:24 AM	Modified By	Shrinidhi Shrinidhi, 9/21/2025, 9:26 AM
- Page Layouts:**

Object	Layout	Assignment
Global	Global Layout [ View Assignment ]	Location Group Assignment Layout [ View Assignment ]
Email Application	Not Assigned [ View Assignment ]	Macro Layout [ View Assignment ]
Home Page Layout	Home Page Default [ View Assignment ]	Object Milestone Layout [ View Assignment ]
- Permissions:**

Users with this profile have the permissions and page layouts listed below. Administrators can change a user's profile by editing that user's personal information.

If your organization uses Record Types, use the Edit links in the Record Type Settings section below to make one or more record types available to users with this profile.

Enabled Permissions:

  - Login IP Ranges (0) | Enabled Apex Class Access (0) | Enabled Visualforce Page Access (0) | Enabled External Data Source Access (0) | Enabled Named Credential Access (0) | Enabled External Credential Principal Access (0) | Enabled Custom Metadata Type Access (0) | Enabled Custom Setting Definitions Access (0) | Enabled Flow Access (0) | Enabled Service Presence Status Access (0) | Enabled Custom Permissions (0)

## 8. Users & Licenses

What I did: I created three test users:

- Alice – Loan Officer with Loan\_Officer\_Profile.
- Bob – Risk Analyst with Risk\_Analyst\_Profile.
- Carol – Manager with Manager\_Profile.

Reason: These test users simulate different roles in the bank. They allow testing of profile permissions, role hierarchy, and data visibility to ensure proper security implementation.

The screenshot shows the Salesforce Setup interface. The left sidebar is titled "Setup" and contains a search bar with "users". Under the "Users" section, "User Management Settings" is expanded, showing "Permission Set Groups", "Permission Sets", "Profiles", "Public Groups", "Queues", "Roles", and "Feature Settings" (which includes "Data.com" and "Prospector Users"). A note at the bottom says "Didn't find what you're looking for? Try using Global Search." The main content area is titled "Users" and shows a "New User" form. The "General Information" section includes fields for First Name (Carol), Last Name (Manager), Alias (cmana), Email (shrinidhishrinidhi452@gmail.com), Username (carol.manager+dev@gmail.com), Nickname (User175847433240087247), Title (empty), Company (empty), Department (empty), Division (empty), Role (Branch Manager), User License (Salesforce Platform), Profile (Standard Platform User), Active (checked), Marketing User (unchecked), Offline User (unchecked), Knowledge User (unchecked), Flow User (unchecked), Service Cloud User (unchecked), Site.com Contributor User (unchecked), and Site.com Publisher User (unchecked). There is a note "I = Required Information" in the top right of the form.

The screenshot shows the Salesforce Setup interface. The left sidebar is identical to the previous one. The main content area is titled "Users" and shows an "Edit" screen for a user named "Bob Risk Analyst". The "General Information" section includes fields for First Name (Bob), Last Name (Risk Analyst), Alias (brisk), Email (shrinidhishrinidhi452@gmail.com), Username (bob.risk+dev@gmail.com), Nickname (User175847399889077236), Title (empty), Company (empty), Department (empty), Division (empty), Role (Risk Analyst), User License (Salesforce), Profile (Risk\_Analyst\_Profile), Active (checked), Marketing User (unchecked), Offline User (unchecked), Knowledge User (unchecked), Flow User (unchecked), Service Cloud User (unchecked), Site.com Contributor User (unchecked), and Site.com Publisher User (unchecked). There is a note "I = Required Information" in the top right of the form.

## 9. Field History Tracking

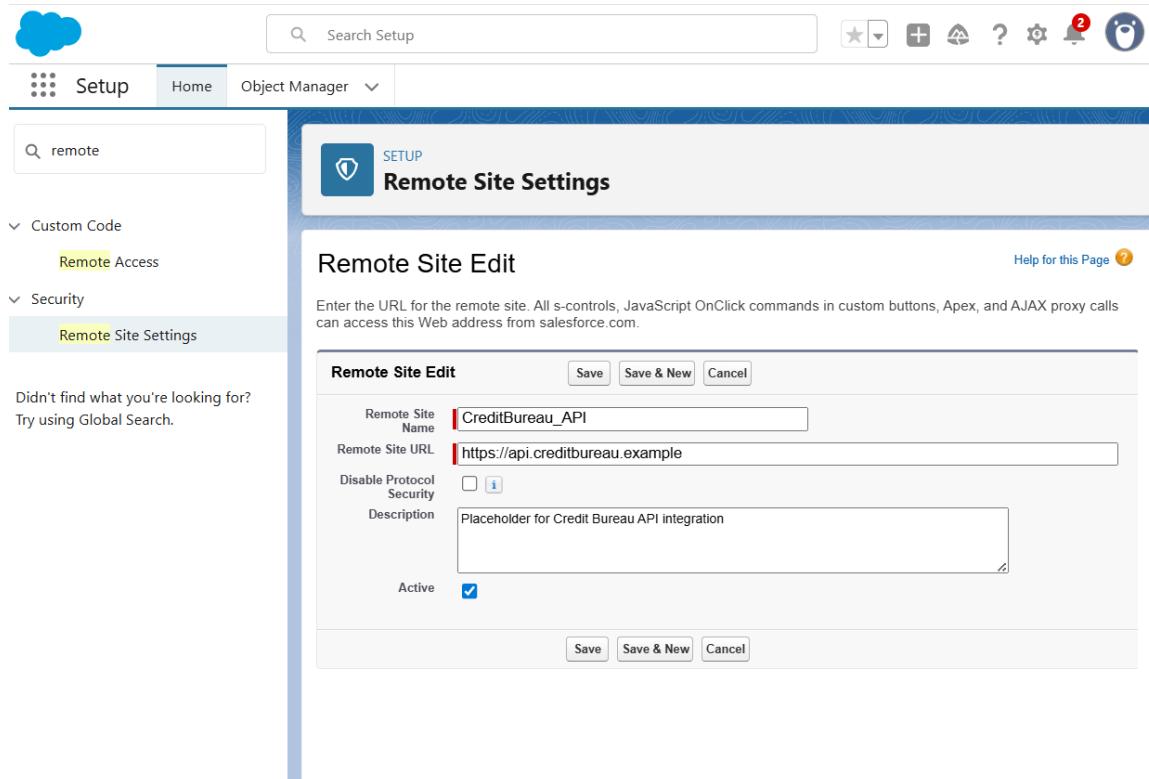
What I did: I enabled history tracking for key fields: Loan\_Status\_\_c, Loan\_Amount\_\_c, Assigned\_To\_\_c (Loan Applications); Payment\_Status\_\_c, Paid\_Amount\_\_c (Repayments); Verification\_Status\_\_c, Expiry\_Date\_\_c (Documents); Offer\_Status\_\_c, Interest\_Rate\_\_c (Offers).

Reason: This ensures an audit trail of critical financial and customer-related fields, supporting compliance, accountability, and transparency.

## 10. Remote Site Settings & Named Credentials

What I did: I added a Remote Site setting for CreditBureau\_API and created a Named Credential (NamedCredential\_CreditBureau).

Reason: Remote Site Settings and Named Credentials are required for secure integrations with external systems like credit bureau APIs, allowing automated credit checks during loan processing.



# Phase 3: Data Modeling & Relationships

In this phase, I created custom objects, fields, record types, page layouts, and relationships to model the FinSmart Bank CRM system. Screenshots are attached for each major step.

## 1. Custom Objects Creation

What I did: I created the following custom objects:

- Bank\_Customer\_\_c (Label: Bank\_Customer)
- Loan\_Application\_\_c (Label: Loan\_Application)
- Repayment\_\_c (Label: Repayment)
- Document\_\_c (Label: Document)
- Offer\_\_c (Label: Offer)

For Loan\_Application\_\_c and Repayment\_\_c, I used Auto-Number record names to ensure unique identifiers (e.g., LA-{0000}, RP-{0000}). For Bank\_Customer\_\_c, I used a text record name to capture full customer names.

- I created custom **tabs** for Bank\_Customer, Loan\_Application, Repayment, Document, and Offer so that users can easily access these objects from the Salesforce app navigation bar.

Reason: These objects represent the core banking entities needed for loan processing, repayments, document tracking, and offers. Using separate objects ensures proper data organization and relationships.

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes the Setup icon, Home button, and Object Manager dropdown. The main header displays 'SETUP > OBJECT MANAGER' and the object name 'Bank\_Customer'. On the left, a sidebar lists various configuration tabs: Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, and Restriction Rules. The right panel is titled 'Details' and contains the following fields:

- Description: [empty]
- API Name: Bank\_Customer\_\_c
- Custom: ✓
- Singular Label: Bank\_Customer
- Plural Label: Bank\_Customers
- Enable Reports: ✓
- Track Activities: ✓
- Track Field History: [empty]
- Deployment Status: Deployed
- Help Settings: [empty]
- Standard salesforce.com Help Window: [empty]

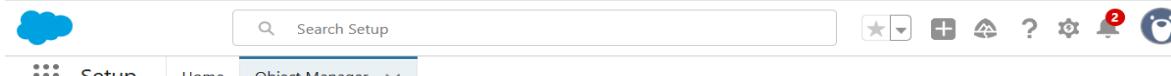
At the bottom right of the details panel are 'Edit' and 'Delete' buttons.



**SETUP > OBJECT MANAGER**

## Loan\_Application

<b>Details</b>	<b>Details</b>
Fields & Relationships	Description
Page Layouts	
Lightning Record Pages	API Name <u>Loan_Application__c</u>
Buttons, Links, and Actions	Custom ✓
Compact Layouts	Singular Label <u>Loan_Application</u>
Field Sets	Plural Label <u>Loan_Applications</u>
Object Limits	
Record Types	
Related Lookup Filters	
Search Layouts	Enable Reports ✓
List View Button Layout	Track Activities ✓
	Track Field History ✓
	Deployment Status <b>Deployed</b>
	Help Settings Standard salesforce.com Help Window



**SETUP > OBJECT MANAGER**

## Repayment

<b>Details</b>	<b>Details</b>
Fields & Relationships	Description
Page Layouts	
Lightning Record Pages	API Name <u>Repayment__c</u>
Buttons, Links, and Actions	Custom ✓
Compact Layouts	Singular Label <u>Repayment</u>
Field Sets	Plural Label <u>Repayments</u>
Object Limits	
Record Types	
Related Lookup Filters	
Search Layouts	Enable Reports ✓
List View Button Layout	Track Activities ✓
	Track Field History
	Deployment Status <b>Deployed</b>
	Help Settings Standard salesforce.com Help Window

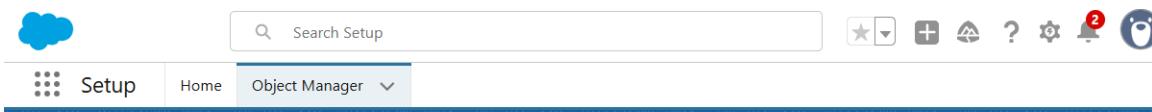


SETUP > OBJECT MANAGER

## Document

Details	
Fields & Relationships	
Page Layouts	
Lightning Record Pages	
Buttons, Links, and Actions	
Compact Layouts	
Field Sets	
Object Limits	
Record Types	
Related Lookup Filters	
Search Layouts	
List View Button Layout	
Restriction Rules	
Description	
API Name	
Document_c	
Custom	
✓	
Singular Label	
Document	
Plural Label	
Documents	
Enable Reports	
✓	
Track Activities	
✓	
Track Field History	
Deployment Status	
Deployed	
Help Settings	
Standard salesforce.com Help Window	

Edit    Delete



SETUP > OBJECT MANAGER

## Offer

Details	
Fields & Relationships	
Page Layouts	
Lightning Record Pages	
Buttons, Links, and Actions	
Compact Layouts	
Field Sets	
Object Limits	
Record Types	
Related Lookup Filters	
Search Layouts	
List View Button Layout	
Restriction Rules	
Description	
API Name	
Offer_c	
Custom	
✓	
Singular Label	
Offer	
Plural Label	
Offers	
Enable Reports	
✓	
Track Activities	
✓	
Track Field History	
Deployment Status	
Deployed	
Help Settings	
Standard salesforce.com Help Window	

Edit    Delete

## 2. Key Fields for Bank\_Customer\_\_c

What I did: I created key fields on Bank\_Customer\_\_c:

- Email\_\_c (Email)
- Mobile\_\_c (Phone)
- Customer\_ID\_\_c (Text, 20)
- DOB\_\_c (Date)
- Address\_\_c (Long Text Area)

Reason: These fields capture essential customer information required for KYC compliance and loan processing.

The screenshot shows the Salesforce Object Manager interface for the 'Bank\_Customer' object. The left sidebar lists various setup options like Page Layouts, Lightning Record Pages, and Field Sets. The main area is titled 'Fields & Relationships' and displays a table of 9 items, sorted by Field Label. The columns are FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The fields listed are: Address (Address\_\_c, Long Text Area(32768)), Created By (CreatedById, Lookup(User)), Customer Name (Name, Text(80)), Customer\_ID (Customer\_ID\_\_c, Text(20) (Unique Case Insensitive)), DOB (DOB\_\_c, Date), Email (Email\_\_c, Email (Unique)), Last Modified By (LastModifiedById, Lookup(User)), Mobile (Mobile\_\_c, Phone), and Owner (OwnerId, Lookup(User,Group)).

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address__c	Long Text Area(32768)		
Created By	CreatedById	Lookup(User)		
Customer Name	Name	Text(80)		
Customer_ID	Customer_ID__c	Text(20) (Unique Case Insensitive)		
DOB	DOB__c	Date		
Email	Email__c	Email (Unique)		
Last Modified By	LastModifiedById	Lookup(User)		
Mobile	Mobile__c	Phone		
Owner	OwnerId	Lookup(User,Group)		

## 3. Key Fields for Loan\_Application\_\_c

What I did: I added key fields on Loan\_Application\_\_c:

- Customer\_\_c (Master-Detail to Bank\_Customer\_\_c)
- Loan\_Type\_\_c (Picklist: Home, Car, Personal, Education, Gold)
- Loan\_Amount\_\_c (Currency)
- Interest\_Rate\_\_c (Percent)
- Tenure\_Months\_\_c (Number)
- EMI\_\_c (Currency, calculated later by Flow/Apex)
- Loan\_Status\_\_c (Picklist: Draft, Submitted, Under\_Review, Approved, Rejected, Disbursed, Active, Closed, Defaulted)
- Credit\_Score\_\_c (Number)
- Assigned\_To\_\_c (Lookup to User)
- Application\_Date\_\_c, Disbursement\_Date\_\_c (Date fields)

Reason: These fields track all key attributes of a loan application, including type, status, amount, repayment terms, and ownership.

**Fields & Relationships**  
27 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Annual_Income	Annual_Income_c	Currency(18, 0)		
Application_Date	Application_Date_c	Date		
Assigned_To	Assigned_To_c	Lookup(User)	✓	
Bank_Customer	Bank_Customer_c	Master-Detail(Bank_Customer)	✓	
Created_By	CreatedById	Lookup(User)		
Credit_Score	Credit_Score_c	Number(18, 0)		
Dealer_Name	Dealer_Name_c	Text(10)		
Down_Payment	Down_Payment_c	Currency(18, 0)		

## 4. Key Fields for Repayment\_c

What I did: I created the following fields on Repayment\_c:

- Loan\_c (Master-Detail to Loan\_Application\_c)
- Installment\_Number\_c (Number)
- Due\_Date\_c, Payment\_Date\_c (Date)
- Due\_Amount\_c, Paid\_Amount\_c (Currency)
- Payment\_Status\_c (Picklist: Due, Paid, Overdue, Partially\_Paid)

Reason: This object allows tracking of repayment schedules, ensuring each installment is monitored accurately.

**Fields & Relationships**  
9 Items, Sorted by Field Label

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created_By	CreatedBy	Lookup(User)		
Due_Amount	Due_Amount_c	Currency(12, 2)		
Due_Date	Due_Date_c	Date		
Installment_Number	Installment_Number_c	Number(18, 0)		
Last_Modified_By	LastModifiedById	Lookup(User)		
Loan	Loan_c	Master-Detail(Loan_Application)	✓	
Paid_Amount	Paid_Amount_c	Currency(12, 2)		
Payment_Status	Payment_Status_c	Picklist		
Repayment_Name	Name	Auto Number	✓	

## 5. Key Fields for Document\_c

What I did: I added the following fields:

- Related\_Loan\_c (Lookup to Loan\_Application\_c)
- Bank\_Customer\_c (Optional Lookup)
- Document\_Type\_c (Picklist: PAN, Aadhaar, Salary Slip, Bank Statement, Address Proof, Other)
- Document\_Number\_c (Text)
- Issue\_Date\_c, Expiry\_Date\_c (Date)
- Verification\_Status\_c (Picklist: Pending, Verified, Rejected)
- Uploaded\_By\_c (Lookup to User)
- Notes\_c (Long Text Area)

Reason: Document\_c stores metadata for KYC and proof documents, while the actual files are managed using Salesforce Files.

The screenshot shows the Salesforce setup interface with the 'Object Manager' for the 'Document' object. The left sidebar lists various setup options like Details, Fields &amp; Relationships, Page Layouts, etc. The main area is titled 'Fields &amp; Relationships' with a sub-header '10 Items, Sorted by Field Label'. It displays a table of fields and their details. The fields listed are: Created By, Customer, Document Name, Document\_Type, Last Modified By, Notes, Owner, and Uploaded\_By. Each field entry includes its name, type, and a detailed description of its relationship (e.g., 'Customer' is a 'Customer\_c' field of type 'Lookup(Bank\_Customer)').

## 6. Key Fields for Offer\_c

What I did: I created fields for Offer\_c:

- Bank\_Customer\_c (Lookup)
- Offer\_Type\_c (Picklist: Personal Loan, Home Loan, Car Loan, Credit Card, Insurance)
- Offer\_Amount\_c (Currency)
- Interest\_Rate\_c (Percent)
- Tenure\_Months\_c (Number)
- Offer\_Status\_c (Picklist: Draft, Sent, Accepted, Rejected, Expired)
- Valid\_From\_c, Valid\_To\_c (Date)
- Generated\_By\_c (Lookup User)
- Response\_Date\_c (Date)
- Notes\_c (Long Text Area)

Reason: Offer\_c manages pre-approved or promotional loan offers, which can be accepted by customers to generate loan applications automatically.

The screenshot shows the Salesforce Setup interface with the Object Manager selected. Under the Offer object, the Fields & Relationships tab is active. The Fields & Relationships section lists 15 items, sorted by Field Label. The fields listed are:

Offer Name	Name	Type
Offer_Amount	Offer_Amount_c	Currency(12, 0)
Offer_Status	Offer_Status_c	Picklist
Offer_Type	Offer_Type_c	Picklist
Owner	OwnerId	Lookup(User,Group)
Response_Date	Response_Date_c	Date
Tenure_Months	Tenure_Months_c	Number(12, 0)
Valid_From	Valid_From_c	Date
Valid_To	Valid_To_c	Date

## 7. Record Types & Page Layouts

What I did: For Loan\_Application\_c, I created Record Types for Home Loan, Vehicle Loan, Personal Loan, and Gold Loan.

Each record type has its own page layout customized with fields relevant to that loan type.

Reason: Record Types allow tailoring loan application forms to specific loan categories, making data entry simpler and more relevant for users.

The screenshot shows the Salesforce Setup interface with the Object Manager selected. Under the Loan\_Application object, the Record Types tab is active. A single record type named "Gold Loan" is displayed. The details for this record type are as follows:

Record Type Label	Gold Loan	Active	
Record Type Name	Gold_Loan	<input checked="" type="checkbox"/>	
Namespace Prefix			
Description			
Created By	Shrinidhi Shrinidhi, 9/22/2025, 4:48 AM	Modified By	Shrinidhi Shrinidhi, 9/22/2025, 4:48 AM

Below the main record type information, there is a table titled "Picklists Available for Editing" which lists two entries:

Action	Field	Modified Date
Edit	Loan_Status	9/22/2025, 4:48 AM
Edit	Loan_Type	9/22/2025, 4:48 AM

Setup > OBJECT MANAGER

## Loan\_Application

**Record Type**

**Personal Loan**

Help for this Page [?](#)

« Back to Custom Object: [Loan\\_Application](#)

Use the Edit button to change the properties of this record type. Use the Edit links in the Picklist Values related list to choose the picklist values available for records with this record type.

Record Type Label	Personal Loan	Active	<input checked="" type="checkbox"/>
Record Type Name	Personal_Loan		
Namespace Prefix			
Description			
Created By	<a href="#">Shrinidhi Shrinidhi</a> , 9/22/2025, 4:48 AM	Modified By	<a href="#">Shrinidhi Shrinidhi</a> , 9/22/2025, 4:48 AM

**Picklists Available for Editing**

Action	Field	Modified Date
Edit	Loan_Status	9/22/2025, 4:48 AM
Edit	Loan_Type	9/22/2025, 4:48 AM

**Picklists Available for Editing Help** [?](#)

Setup > OBJECT MANAGER

## Loan\_Application

**Record Type**

**Home Loan**

Help for this Page [?](#)

« Back to Custom Object: [Loan\\_Application](#)

Use the Edit button to change the properties of this record type. Use the Edit links in the Picklist Values related list to choose the picklist values available for records with this record type.

Record Type Label	Home Loan	Active	<input checked="" type="checkbox"/>
Record Type	Record Type: Home Loan ~ Salesforce - Developer Edition		
Namespace Prefix			
Description			
Created By	<a href="#">Shrinidhi Shrinidhi</a> , 9/22/2025, 4:45 AM	Modified By	<a href="#">Shrinidhi Shrinidhi</a> , 9/22/2025, 4:45 AM

**Picklists Available for Editing**

Action	Field	Modified Date
Edit	Loan_Status	9/22/2025, 4:45 AM
Edit	Loan_Type	9/22/2025, 4:45 AM

**Picklists Available for Editing Help** [?](#)

The screenshot shows the Salesforce Setup interface with the following details:

**Header:** Search Setup, Home, Object Manager

**Breadcrumbs:** SETUP > OBJECT MANAGER

**Section:** **Loan\_Application**

**Left Sidebar (Record Types):**

- Details
- Fields & Relationships
- Page Layouts
- Lightning Record Pages
- Buttons, Links, and Actions
- Compact Layouts
- Field Sets
- Object Limits
- Record Types** (selected)
- Related Lookup Filters
- Search Layouts
- List View Button Layout

**Record Type Page:**

**Record Type:** **Vehicle Loan**

**Description:** Use the Edit button to change the properties of this record type. Use the Edit links in the Picklist Values related list to choose the picklist values available for records with this record type.

**Record Type Details:**

Record Type Label	Vehicle Loan	Active	<input checked="" type="checkbox"/>
Record Type Name	Vehicle_Loan		
Namespace Prefix			
Description			
Created By	Shrinidhi Shrinidhi, 9/22/2025, 4:46 AM	Modified By	Shrinidhi Shrinidhi, 9/22/2025, 4:46 AM

**Picklists Available for Editing:**

Action	Field	Modified Date
Edit	Loan_Status	9/22/2025, 4:46 AM
Edit	Loan_Type	9/22/2025, 4:46 AM

**Help:** Help for this Page [?](#)

## Page Layouts:

The screenshot shows the Salesforce Setup interface with the following details:

**Header:** Search Setup, Home, Object Manager

**Breadcrumbs:** SETUP > OBJECT MANAGER

**Section:** **Loan\_Application**

**Left Sidebar (Page Layouts):**

- Details
- Fields & Relationships
- Page Layouts** (selected)
- Lightning Record Pages
- Buttons, Links, and Actions
- Compact Layouts
- Field Sets
- Object Limits
- Record Types
- Related Lookup Filters
- Search Layouts
- List View Button Layout

**Page Layouts Page:**

**Section:** **Page Layouts**  
5 Items, Sorted by Page Layout Name

**Actions:** Quick Find, New, Page Layout Assignment

**Table:** Page Layouts

PAGE LAYOUT NA...	CREATED BY	MODIFIED BY
Gold Loan	Shrinidhi Shrinidhi, 9/22/2025, 5:24 AM	Shrinidhi Shrinidhi, 9/22/2025, 5:25 AM
Home Loan	Shrinidhi Shrinidhi, 9/22/2025, 5:12 AM	Shrinidhi Shrinidhi, 9/22/2025, 5:24 AM
Loan_Application Layout	Shrinidhi Shrinidhi, 9/20/2025, 6:59 AM	Shrinidhi Shrinidhi, 9/22/2025, 5:24 AM
Personal Loan	Shrinidhi Shrinidhi, 9/22/2025, 5:19 AM	Shrinidhi Shrinidhi, 9/22/2025, 5:24 AM
Vehicle Loan	Shrinidhi Shrinidhi, 9/22/2025, 5:18 AM	Shrinidhi Shrinidhi, 9/22/2025, 5:24 AM

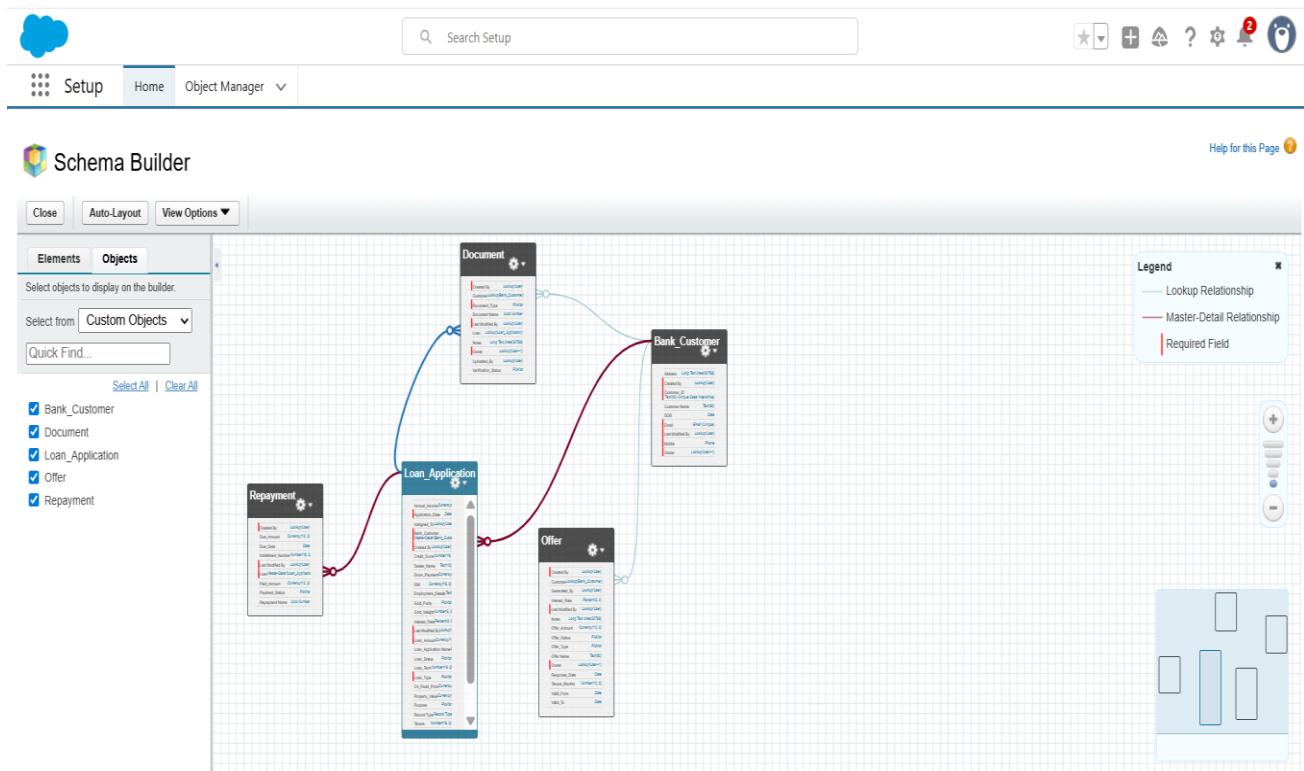
## 9. Schema Builder

What I did: I used Schema Builder to visualize relationships:

Bank\_Customer\_\_c (1) → Loan\_Application\_\_c (N) → Repayment\_\_c (N)

Additionally, Loan\_Application\_\_c is related to Document\_\_c and Offer\_\_c.

Reason: Schema Builder provides a clear picture of how objects are connected, ensuring proper data relationships are maintained.



# Phase 4: Process Automation (Admin)

Validation Rules, Scheduled Flow, and Approval Process

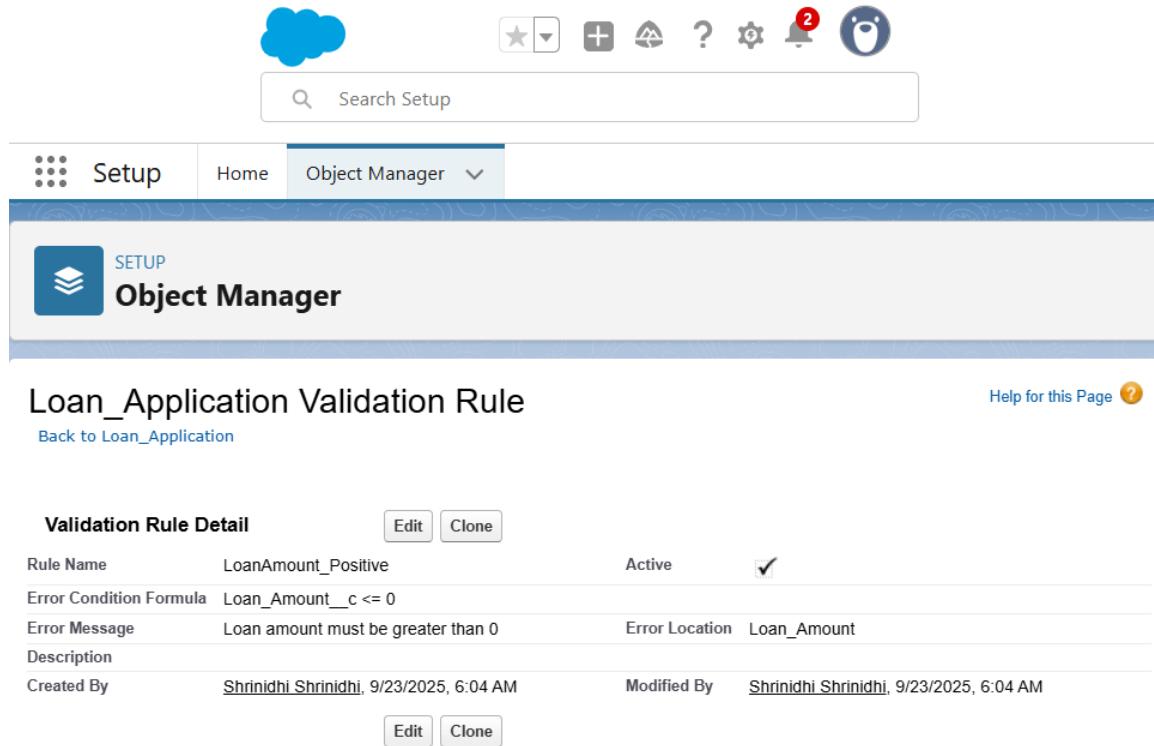
## 1. Validation Rule — Loan Amount > 0

What I did: I created a validation rule on Loan\_Application\_\_c object to ensure that Loan Amount is always greater than 0.

- Rule Name: LoanAmount\_Positive
- Formula: Loan\_Amount\_\_c <= 0
- Error Message: Loan amount must be greater than 0.
- Error Location: Loan\_Amount\_\_c field.

Reason: This prevents users from entering zero or negative loan amounts, which are invalid in real-world banking operations.

Testing: Tried creating a Loan with Loan\_Amount\_\_c = 0 → System displayed the error message.



The screenshot shows the Salesforce Object Manager interface. At the top, there's a navigation bar with icons for Home, Setup, Object Manager (selected), and other setup options. Below the navigation bar is a search bar labeled "Search Setup". The main area is titled "Object Manager" with a "SETUP" button. Underneath, the title "Loan\_Application Validation Rule" is displayed, along with a "Help for this Page" link. A "Back to Loan\_Application" link is also present. The "Validation Rule Detail" section contains the following information:

Validation Rule Detail		Edit	Clone
Rule Name	LoanAmount_Positive	Active	<input checked="" type="checkbox"/>
Error Condition Formula	Loan_Amount__c <= 0		
Error Message	Loan amount must be greater than 0	Error Location	Loan_Amount
Description			
Created By	Shrinidhi Shrinidhi, 9/23/2025, 6:04 AM		
<a href="#">Edit</a> <a href="#">Clone</a>			

## 2. Scheduled Flow — EMI Reminder Emails

What I did: I built a Schedule-Triggered Flow to send EMI reminder emails to customers 3 days before due date.

Steps:

### 1. Created an Email Template named Loan Repayment Reminder.

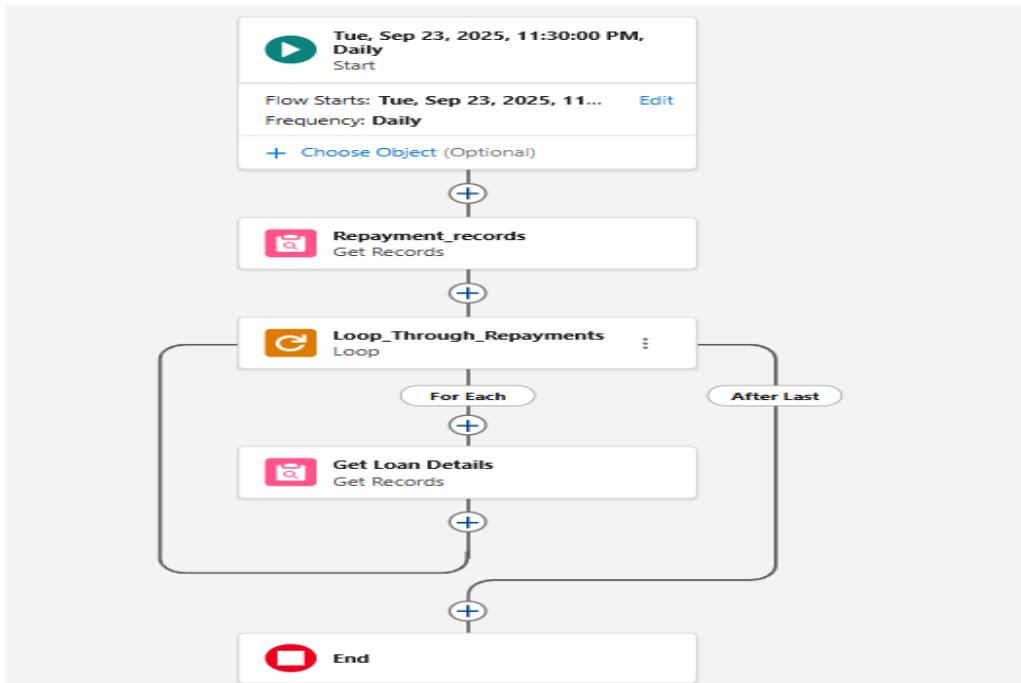
The screenshot shows the Salesforce Email Template page for 'Loan Repayment Reminder'. The page has a header with navigation links like Sales, Home, Opportunities, Leads, Tasks, Files, Accounts, Contacts, Campaigns, Dashboards, Reports, Chatter, and a search bar. Below the header, the template name 'Loan Repayment Reminder' is displayed. The main content area is divided into sections: 'Information' and 'Message Content'. In the 'Information' section, fields include 'Email Template Name' (Loan Repayment Reminder), 'Related Entity Type' (Repayment), 'Description' (template for sending EMI due reminders to customers 3 days before repayment date), and 'Folder' (Loan Notifications). In the 'Message Content' section, there are two parts: 'Subject' (FinSmart: EMI reminder for Loan {{RelatedTo.Loan\_r.Name}}) and 'HTML Value'. The 'HTML Value' part contains a message to the customer about their EMI reminder, including the amount and due date, followed by a note about paying on time to avoid penalties, and a closing thanks from FinSmart Bank.

### 2. Built a Scheduled Flow:

- Frequency: Daily
- Formula Resource: `targetDate = $Flow.CurrentDate + 3`
- Get Records: `Repayment__c` where `Due__Date__c = targetDate` and `Payment_Status__c = 'Due'`
- Loop through records and send email using Send Email action.

Reason: This ensures that customers are reminded about their EMI payments on time, reducing the chance of defaults and penalties.

Testing: For testing, I set `targetDate = $Flow.CurrentDate + 0`. Emails successfully triggered to test records.



### 3. Approval Process — High Value Loan Approvals

What I did: I created an Approval Process for Loan Applications where Loan Amount is  $\geq 1,000,000$ .

Steps:

1. Object: Loan\_Application\_c
2. Entry Criteria: Loan\_Amount\_\_c  $\geq 1000000$
3. Approver: Manager of the Loan Owner (or specified role)
4. Initial Submission Action:
  - Field Update: Loan\_Status\_\_c = 'Under\_Review'
  - Optional: Email alert to approver
5. Approval Steps: Approver reviews and approves/rejects
6. Final Approval Actions:
  - Field Update: Loan\_Status\_\_c = 'Approved'
  - Optional: Notification to submitter
  - Trigger repayment creation process (via flow)
7. Final Rejection Actions:
  - Field Update: Loan\_Status\_\_c = 'Rejected'
  - Email notification to applicant

Reason: This ensures that high-value loans go through a manager's approval before being processed, adding control and reducing risk.

Testing: Created a loan with Loan\_Amount\_\_c = 1,200,000. Submitted for Approval → Manager received request → Approved → Loan\_Status\_\_c updated to 'Approved'.

The screenshot shows the Salesforce Setup interface. The left sidebar has a search bar and navigation links for Home, Object Manager, and various setup categories like Data, Feature Settings, Process Automation, and Approval Processes. A message at the bottom says "Didn't find what you're looking for? Try using Global Search." The main content area is titled "Approval Processes" and shows a specific process named "Loan\_Application: Loan Approval > 1M". The process definition details include:

Process Name	Loan Approval > 1M	Active	<input type="checkbox"/>	Approval Processes: Loan_Approval_1M
Unique Name	Loan_Approval_1M	Next Automated Approver Determined By	Manager of Record Submitter	
Description				
Entry Criteria	Loan_Application: Loan_Amount GREATER OR EQUAL 100000			
Record Editability	Administrator OR Current Approver	Allow Submitters to Recall Approval Requests	<input type="checkbox"/>	
Approval Assignment Email Template	Approval_Email			
Initial Submitters	Role: Loan Officer			
Created By	Shrinidhi Shrinidhi, 9/24/2025, 12:14 AM		Modified By	Shrinidhi Shrinidhi, 9/24/2025, 12:19 AM

Below the details is a section for "Initial Submission Actions" with a table:

Action	Type	Description
Record Lock		Lock the record from being edited

## Summary

- Validation Rule ensures data accuracy (Loan Amount > 0).
- Scheduled Flow ensures timely communication (EMI reminders).
- Approval Process ensures governance and risk control for high-value loans.

# Phase 5: Apex Programming (Developer)

## 1: Create a Simple Trigger (Loan Application Auto Status)

- Purpose: Auto-set Loan\_Status\_\_c when a loan is created.
- What I did: Created an Apex Trigger on Loan\_Application\_\_c object to automatically set status based on Loan Amount.
- Reason: Ensures loan applications are automatically categorized as 'Draft' or 'Under\_Review' depending on loan amount, reducing manual work and enforcing business rules.

**Code:**

```
trigger LoanApplicationTrigger on Loan_Application__c (before insert) {  
    for (Loan_Application__c loan : Trigger.new) {  
        if (loan.Loan_Amount__c != null) {  
            if (loan.Loan_Amount__c >= 500000) {  
                loan.Loan_Status__c = 'Under_Review';  
            } else {  
                loan.Loan_Status__c = 'Draft';  
            }  
        }  
    }  
}
```

```
1 trigger LoanApplicationTrigger on Loan_Application__c (before insert)
2     for (Loan_Application__c loan : Trigger.new) {
3         if (loan.Loan_Amount__c != null) {
4             if (loan.Loan_Amount__c >= 500000) {
5                 loan.Loan_Status__c = 'Under_Review';
6             } else {
7                 loan.Loan_Status__c = 'Draft';
8             }
9         }
10    }
11 }
```

## 2: Create a Simple Apex Class (Helper for EMI)

- Purpose: Calculate EMI (basic formula).
- What I did: Created an Apex Class LoanHelper to calculate EMI values.
- Reason: Provides a reusable utility to compute EMI, useful for loan simulations and repayment planning.

### Code:

```
public with sharing class LoanHelper {  
    public static Decimal calculateEMI(Decimal amount, Decimal annualRate, Integer months) {  
        if (amount == null || annualRate == null || months == null || months <= 0) {  
            return 0;  
        }  
        // Convert to Double for pow()  
        Double principal = amount.doubleValue();  
        Double monthlyRate = (annualRate.doubleValue() / 12) / 100;  
        Double n = Double.valueOf(months);
```

```

// Formula: EMI = P * r * (1+r)^n / ((1+r)^n - 1)

Double powVal = Math.pow(1 + monthlyRate, n);

Double emiDouble = (principal * monthlyRate * powVal) / (powVal - 1);

// Cast back to Decimal

Decimal emi = Decimal.valueOf(emiDouble);

return emi.setScale(2); // round to 2 decimal places

}
}

```

### 3: Simulate Credit Bureau Integration

- Purpose: Show external API call for Credit Score.
- What I did: Created a Named Credential, Apex Service class, and Mock class for integration testing.
- Reason: Simulates integration with external systems like Credit Bureau, demonstrating how Salesforce can handle callouts, responses, and use them to update loan application status.

#### 3.1 Apex Class to Call Credit Bureau:

```

public with sharing class CreditBureauService {
    @future(callout=true)

    public static void getCreditScore(Id loanId) {
        Loan_Application__c loan = [SELECT Id, Credit_Score__c FROM Loan_Application__c
WHERE Id = :loanId LIMIT 1];

```

```

        HttpRequest req = new HttpRequest();
        req.setEndpoint('callout:NamedCredential_CreditBureau/score?cust=' + loan.Id);
        req.setMethod('GET');

```

```

        Http http = new Http();
        HttpResponse res = http.send(req);

```

```

        if (res.getStatusCode() == 200) {

            Integer score = Integer.valueOf(res.getBody());

            loan.Credit_Score__c = score;

            if (score >= 650) loan.Loan_Status__c = 'Under_Review';

        else loan.Loan_Status__c = 'Rejected';

            update loan;

        }

    }

}

```

### **3.2 Mock Class for Testing:**

```

@IsTest

global class CreditBureauMock implements HttpCalloutMock {

    global HttpResponse respond(HttpRequest req) {

        HttpResponse res = new HttpResponse();

        res.setStatusCode(200);

        res.setBody('720'); // fake credit score

        return res;

    }

}

```

## **4: Create Test Classes**

- Purpose: Ensure >75% coverage and validate logic.
- What I did: Wrote test classes for LoanHelper and CreditBureauService to validate functionality and meet Salesforce coverage requirements.
- Reason: Test classes are mandatory in Salesforce to deploy Apex code and ensure system reliability.

### **LoanHelperTest.cls:**

```

@isTest

private class LoanHelperTest {

    @isTest

```

```

static void testLoanApplicationTriggerAndEMI() {
    Bank_Customer__c cust = new Bank_Customer__c(Name='Test Cust',
Email__c='test@test.com');

    insert cust;

    Loan_Application__c loan = new Loan_Application__c(
        Bank_Customer__c = cust.Id,
        Loan_Amount__c = 600000,
        Interest_Rate__c = 10,
        Tenure_Months__c = 12
    );
    insert loan; // trigger runs

    Loan_Application__c inserted = [SELECT Loan_Status__c FROM Loan_Application__c
WHERE Id=:loan.Id];

    System.assertEquals('Under_Review', inserted.Loan_Status__c);

    Decimal emi = LoanHelper.calculateEMI(100000, 12, 12);

    System.assert(emi > 0, 'EMI should be positive');
}
}

```

### **CreditBureauServiceTest.cls:**

```

@isTest
private class CreditBureauServiceTest {
    @isTest
    static void testCreditScoreUpdate() {
        Bank_Customer__c cust = new Bank_Customer__c(Name='Cust B',
Email__c='cust@test.com');

```

```
insert cust;

Loan_Application__c loan = new Loan_Application__c(
    Bank_Customer__c = cust.Id,
    Loan_Amount__c = 400000,
    Loan_Status__c = 'Submitted'
);

insert loan;

// mock callout
Test.setMock(HttpCalloutMock.class, new CreditBureauMock());

Test.startTest();
CreditBureauService.getCreditScore(loan.Id);
Test.stopTest();

Loan_Application__c updated = [SELECT Credit_Score__c, Loan_Status__c FROM
    Loan_Application__c WHERE Id = :loan.Id];
System.assertEquals(720, updated.Credit_Score__c);
System.assertEquals('Under_Review', updated.Loan_Status__c);
}
```

## 5: Simple Scheduled Apex

- Purpose: Simple scheduled check for overdue repayments.
- What I did: Created a schedulable class that fetches overdue repayments and logs them.
- Reason: Demonstrates ability to schedule background jobs in Salesforce to monitor repayment compliance.

### Code:

```
public with sharing class OverdueRepaymentReminder implements Schedulable {  
    public void execute(SchedulableContext sc) {  
        List<Repayment__c> overdues = [  
            SELECT Id, Loan__c, Due_Amount__c  
            FROM Repayment__c  
            WHERE Payment_Status__c = 'Overdue'  
        ];  
        for (Repayment__c r : overdues) {  
            System.debug('Overdue repayment: ' + r.Id + ' Amount: ' + r.Due_Amount__c);  
        }  
    }  
}
```

To schedule: Setup → Apex Classes → Schedule Apex → Job Name = OverdueReminder, Class = OverdueRepaymentReminder, set time.

The screenshot shows the Salesforce Setup interface for managing Apex Classes. The left sidebar has a search bar and navigation links for Home and Object Manager. Under Custom Code, the Apex Classes link is selected. A message says "Didn't find what you're looking for? Try using Global Search." The main content area is titled "Apex Classes" and shows the "Schedule Apex" section. It prompts to "Schedule an Apex class that implements the Schedulable interface to be automatically executed on a specified interval." The form includes fields for "Job Name" (OverdueReminder), "Apex Class" (OverdueRepaymentRemind), and "Schedule Using" (Schedule Builder). The "Frequency" section is set to "Weekly". To the right, a sidebar lists days of the week with checkboxes: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, and Saturday, all of which are checked.

Search Setup

Setup Home Object Manager

Apex class

Custom Code

Apex Classes

Didn't find what you're looking for?  
Try using Global Search.

Help for this Page ?

SETUP Apex Classes

Schedule Apex

Schedule an Apex class that implements the Schedulable interface to be automatically executed on a specified interval.

Job Name: OverdueReminder

Apex Class: OverdueRepaymentRemind

Schedule Using: Schedule Builder

Cron Expression

Schedule Apex Execution

Frequency: Weekly

Monthly

Recur every week on:

- Sunday
- Monday
- Tuesday
- Wednesday
- Thursday
- Friday
- Saturday

# Phase 6: User Interface Development

## 1: Create Lightning App

What I did: I created a new Lightning App using App Manager in Salesforce to organize all loan-related features in one application.

Reason: A Lightning App provides a dedicated workspace for users. By grouping related objects and features (Customers, Loan Applications, Repayments, Documents, Reports), it improves navigation and user experience.

Steps:

1. Go to Setup → App Manager → New Lightning App.
2. Enter details:
  - App Name: FinSmart Banking App
  - Navigation Type: Standard (Lightning)
  - Optionally upload a logo → Next
3. Add Tabs:
  - Customers
  - Loan Applications
  - Repayments
  - Documents
  - Reports
4. Save & Finish.

New Lightning App

App Details	App Branding
<p>* App Name <small>i</small></p> <input type="text" value="FinSmart Banking App"/> <p>* Developer Name <small>i</small></p> <input type="text" value="FinSmart_Banking_App"/>	<p>Image <small>i</small></p> <div style="border: 1px solid #ccc; padding: 10px; text-align: center;"><input type="button" value="Upload"/></div> <p>Primary Color Hex Value <small>i</small></p> <div style="display: flex; align-items: center;"><span style="background-color: red; width: 10px; height: 10px; margin-right: 10px;"></span><input type="text" value="#D7330E"/></div>
<p>Description <small>i</small></p> <div style="border: 1px solid #ccc; padding: 5px; height: 40px; margin-top: 10px;">Enter a description...</div>	<p>Org Theme Options</p> <p><input type="checkbox"/> Use the app's image and color instead of the org's custom theme</p>
<p>App Launcher Preview</p> <div style="border: 1px solid #ccc; width: 100%; height: 20px; margin-top: 10px;"></div>	
<p>Next</p>	

## New Lightning App

Available Items [Create ▾](#)

Type to filter list...

- Accounts
- Activation Targets
- Activations
- All Sites
- Alternative Pay...
- Analytics
- App Launcher

Selected Items

- Bank...
- Loan...
- Repa...
- Docu...
- Offers

Back Next

## New Lightning App

### User Profiles

Choose the user profiles that can access this app.

Available Profiles

Standar ...

Customer Portal M...

Standard Platform ...

Selected Profiles

Loan\_Officer\_Profile

Risk\_Analyst\_Profile

Manager\_Profile

System Administrat...

Standard User

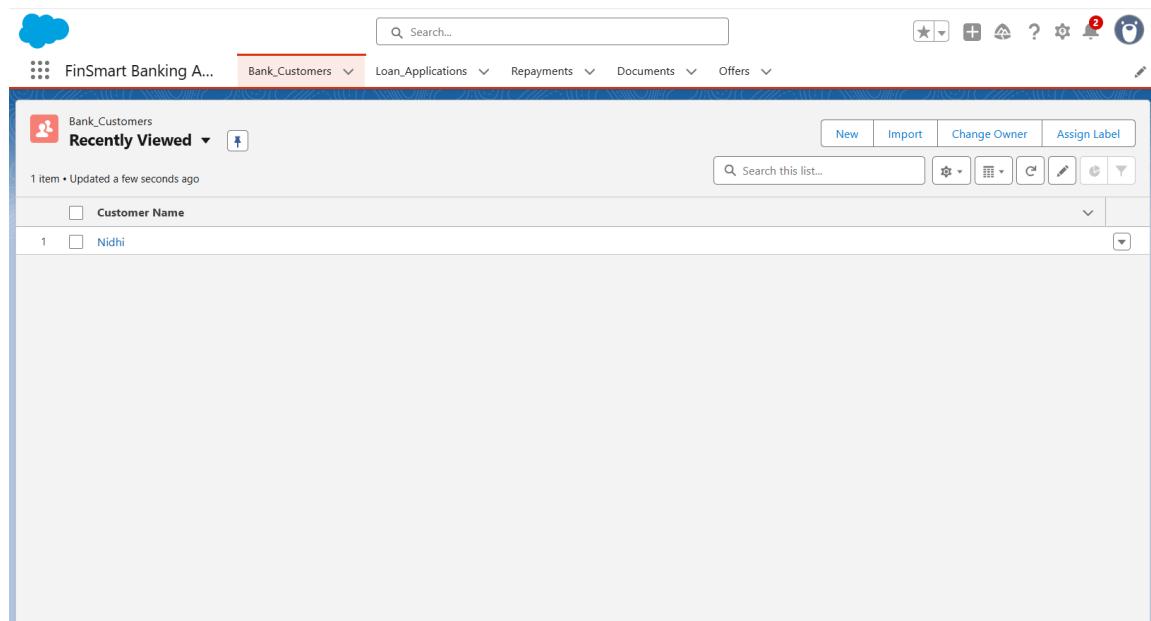
## Step 2: Confirm Tabs in App Launcher

What I did: I verified the app setup by launching it from the App Launcher and ensuring all required tabs are visible.

Reason: Verification ensures the app is properly configured and that users will be able to access all necessary components in their daily work.

Steps:

1. Go to Setup → App Launcher.
2. Open the newly created FinSmart Banking App.
3. Confirm that all required tabs (Customers, Loan Applications, Repayments, Documents, Reports) are visible.



## Summary

- What I did: Created a dedicated Lightning App named FinSmart Banking App and added navigation tabs for core objects.
- Reason: To provide a centralized, easy-to-use application that helps users manage customers, loans, repayments, documents, and reports efficiently.

Next step: Add screenshots of app creation and tab confirmation for documentation.

# Phase 7: Integration & External Access

Credit Bureau Integration

## Step 1: Named Credential

What I did: I created a Named Credential in Salesforce for connecting to the Credit Bureau API.

Reason: Named Credentials simplify API callouts by securely storing the endpoint URL and authentication details. This avoids hardcoding sensitive information in Apex classes and makes future updates easier.

Steps:

1. Go to Setup → Named Credentials → New.

The screenshot shows the Salesforce Setup interface. The top navigation bar includes a blue cloud icon, a search bar labeled 'Search Setup', and various navigation icons. Below the bar, the 'Setup' tab is selected, followed by 'Home' and 'Object Manager'. A sidebar on the left is titled 'Security' and contains a 'Named Credentials' section, which is highlighted with a yellow background. The main content area displays a 'SETUP > NAMED CREDENTIALS' page for a credential named 'NamedCredential\_CreditBureau'. The page has three tabs: 'Edit' and 'Delete' on the right, and a 'Label' field containing 'NamedCredential\_CreditBureau'. The 'Name' field also contains 'NamedCredential\_CreditBureau'. Below these fields are 'URL' and 'Enabled for Callouts' settings. The 'Authentication' section lists an 'External Credential' named 'CreditBureau\_ExtCredential'. The 'Callout Options' section includes 'Generate Authorization Header' (which is checked) and 'Allow Formulas in HTTP Header'.

### Apex Callout:

```
public with sharing class CreditBureauService {  
    @future(callout=true)
```

```

public static void getCreditScore(Id loanId) {

    Loan_Application__c loan = [SELECT Id, Credit_Score__c FROM Loan_Application__c
WHERE Id = :loanId LIMIT 1];

    HttpRequest req = new HttpRequest();
    req.setEndpoint('callout:NamedCredential_CreditBureau/score?cust=' + loan.Id);
    req.setMethod('GET');

    Http http = new Http();
    HttpResponse res = http.send(req);

    if (res.getStatusCode() == 200) {
        Integer score = Integer.valueOf(res.getBody());
        loan.Credit_Score__c = score;
        if (score >= 650) loan.Loan_Status__c = 'Under_Review';
        else loan.Loan_Status__c = 'Rejected';
        update loan;
    }
}
}

```

## Step 2: Test Callout (Using HttpCalloutMock)

What I did: I created a Mock class and a Test class to simulate the Credit Bureau API response without calling a real service.

Reason: Salesforce requires test classes to mock callouts. This ensures code can be deployed while avoiding dependencies on external systems during testing.

### Mock Class Code:

@IsTest

```
global class CreditBureauMock implements HttpCalloutMock {
```

```

global HttpResponse respond(HttpServletRequest req) {

    HttpServletResponse res = new HttpServletResponse();
    res.setStatusCode(200);
    res.setBody('720'); // fake credit score
    return res;
}

}

```

**Test Class Code:**

```

@isTest
private class CreditBureauServiceTest {

    @isTest
    static void testCreditScoreUpdate() {
        Bank_Customer__c cust = new Bank_Customer__c(Name='Cust B',
Email__c='cust@test.com');

        insert cust;

        Loan_Application__c loan = new Loan_Application__c(
            Bank_Customer__c = cust.Id,
            Loan_Amount__c = 400000,
            Loan_Status__c = 'Submitted'
        );
        insert loan;

        // mock callout
        Test.setMock(HttpCalloutMock.class, new CreditBureauMock());

        Test.startTest();
    }
}

```

```
CreditBureauService.getCreditScore(loan.Id);

Test.stopTest();

Loan_Application__c updated = [SELECT Credit_Score__c, Loan_Status__c FROM
Loan_Application__c WHERE Id = :loan.Id];

System.assertEquals(720, updated.Credit_Score__c);

System.assertEquals('Under_Review', updated.Loan_Status__c);

}

}
```

## Summary

- What I did: Configured a Named Credential and implemented a minimal Apex callout setup with test classes.
- Reason: This ensures Salesforce can simulate Credit Bureau integration securely and pass deployment requirements without relying on a real API.

Next step: Add screenshots of Named Credential setup and test execution for documentation

# Phase 8: Data Management & Deployment

Data Import & Duplicate Management (Step by Step Procedure)

## Preparation — Create External ID Fields

What I did: I created External ID fields on Customer\_\_c to ensure clean parent-child relationships during CSV imports.

Reason: External ID fields allow Salesforce to match and link records using unique values from CSVs. This is important when importing data with relationships (e.g., Loans linked to Customers).

Steps:

1. Setup → Object Manager → Customer\_\_c → Fields & Relationships → New → Text (Length 20).
2. Label: Customer ID, Field Name: Customer\_ID\_\_c.
3. Mark as External ID and Unique (optional) → Save.

## 1: Import Data using Data Import Wizard

What I did: I used Salesforce Data Import Wizard to upload Customer data from CSV.

Reason: The Data Import Wizard is simple, user-friendly, and ideal for small datasets. It automatically maps fields when column headers match API field names.

Steps:

1. Setup → Data Import Wizard → Launch Wizard.
2. For Customers:
  - Select Custom Objects → choose Customer (Customer\_\_c).
  - Upload Customer.csv.
  - Map fields if required (auto-map works when names match).
  - Start Import → Wait → Review results.

Getting closer...

Choose data      Edit mapping      Start import

### Import your Data into Salesforce

You can import up to 50,000 records at a time.

What kind of data are you importing?

Standard objects     Custom objects

Bank\_Customers

Documents

Loan\_Applications

Offers

Rewards

What do you want to do?

Add new records

Match by:   
Customer\_ID (External ID)

Which User field in your file designates record owners?   
-None-

Trigger workflow rules and processes?   
 Trigger workflow rules and processes for new and updated records

Update existing records

Where is your data located?

Drag CSV file here to upload

CSV

File   
Choose File Customer.csv

Character Code   
ISO-8859-1 (General US & Western European, ISO-LATIN-1)

Values Separated By   
Comma

Cancel Previous Next

Almost done

Choose data      Edit mapping      Start import

### Edit Field Mapping: Bank\_Customers

Your file has been auto-mapped to existing Salesforce fields, but you can edit the mappings if you wish. Unmapped fields will not be imported.

Edit	Mapped Salesforce Object	CSV Header	Example	Example	Example
Change	Customer Name	Name	Rahul Kumar	Priya Sharma	Amit Patel
Change	Customer_ID	Customer_ID_c	CUST001	CUST002	CUST003
Change	Email	Email_c	rahul.kumar@example.com	priya.sharma@example.com	amit.patel@example.com
Change	Mobile	Mobile_c	9876543210	9123456780	9988776555
Change	DOB	DOB_c	1990-05-10	1988-08-20	1985-02-14
Change	Address	Address_c	123 MG Road, Bar...	45 Park Street, Kol...	9 Marine Drive, Mumbai

Cancel Previous Next

A	B	C	D	E	F	G	H	I	J	K	L
1	Id	Success	Created	Error							
2	a00gLO00C	TRUE	TRUE								
3	a00gLO00C	TRUE	TRUE								
4	a00gLO00C	TRUE	TRUE								
5	a00gLO00C	TRUE	TRUE								
6	a00gLO00C	TRUE	TRUE								

## 2: Duplicate Management

What I did: I created Matching Rules and Duplicate Rules to prevent duplicate customer records.

Reason: Duplicate management ensures data quality by preventing multiple entries for the same customer based on email or mobile number.

Steps:

1. Setup → Duplicate Management → **Matching Rules**.

- Customer by Email: New Matching Rule → Object: Customer → add field Email\_c (fuzzy/exact match) → Save & Activate.

The screenshot shows the Salesforce Setup interface for creating a Matching Rule. The left sidebar shows 'Data' and 'Duplicate Management' selected. The main area is titled 'Matching Rules' and displays the 'Rule Details' and 'Matching Criteria' sections. In 'Rule Details', the Object is 'Bank\_Customer', Rule Name is 'Customer by Email', and Unique Name is 'Customer\_by\_Email'. In 'Matching Criteria', there are five fields listed: 'Email' (Exact), '--None--' (Exact), '--None--' (Exact), '--None--' (Exact), and '--None--' (Exact). The 'Matching Method' dropdown for each field is set to 'Exact'. The 'Match Blank Fields' section shows 'AND' selected for all fields. At the bottom right are 'Previous', 'Save', and 'Cancel' buttons.

- Customer by Mobile: New Matching Rule → Object: Customer → add field Mobile\_\_c → Save & Activate.

The screenshot shows the Salesforce Setup interface with the following details:

- Rule Details:**
  - Object: Bank\_Customer
  - Rule Name: Customer by Mobile
  - Unique Name: Customer\_by\_Mobile
  - Description: (empty)
- Matching Criteria:**
  - Field: Mobile
  - Matching Method: Exact
  - Match Blank Fields: AND (checkboxes checked)

## 2. Setup → Duplicate Rules → New Rule for Customer.

- Choose Object: Customer\_\_c.
- Use matching rules above (Email, Mobile).
- Action: Block or Alert when duplicates detected.
- Activate.

3. When re-importing data, duplicates are flagged.

The screenshot shows the Salesforce Setup interface. The top navigation bar includes a cloud icon, a search bar labeled "Search Setup", and various global buttons. The main menu bar has "Setup" selected, followed by "Home" and "Object Manager". A sidebar on the left is titled "Data" and contains "Duplicate Management" and "Duplicate Rules", with "Duplicate Rules" currently selected. A message in the sidebar says, "Didn't find what you're looking for? Try using Global Search." The main content area is titled "d SETUP Duplicate Rules" and displays a "Bank\_Customer Duplicate Rule" named "Customer\_Duplicate\_Check". The rule details are as follows:

Duplicate Rule Detail		Operations On Create		Operations On Edit	
Rule Name	Customer_Duplicate_Check	Order	1 of 1 [ Reorder ]		
Description					
Object	Bank_Customer				
Record-Level Security	Enforce sharing rules				
Action On Create	Block	<input type="checkbox"/> Alert <input type="checkbox"/> Report			
Action On Edit	Block	<input type="checkbox"/> Alert <input type="checkbox"/> Report			
Alert Text	Duplicate				
Active	✓				
Matching Rule	<input checked="" type="checkbox"/> Customer by Email <input checked="" type="checkbox"/> Mapped	Matching Criteria		Bank_Customer: Email EXACT MatchBlank = FALSE	
Conditions					
Created By	Shrinidhi Shrinidhi, 9/24/2025, 8:02 AM	Modified By	Shrinidhi Shrinidhi, 9/24/2025, 8:02 AM		

At the bottom of the detail page are "Edit", "Delete", "Clone", and "Deactivate" buttons.

# Phase 9: Reporting, Dashboards & Security Review

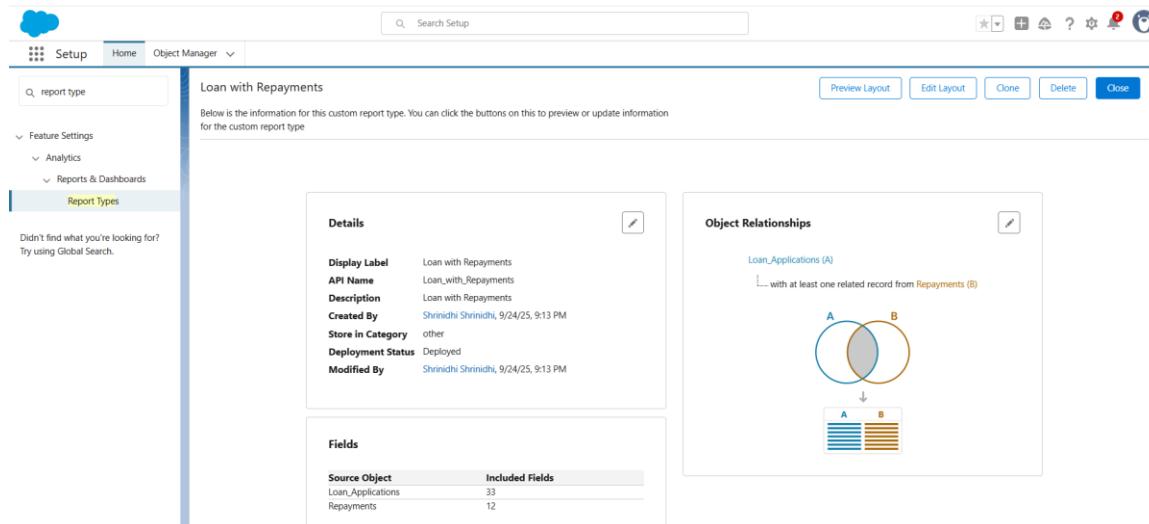
## Step 1: Create Custom Report Types

What I did: I created custom report types to combine related objects for reporting.

Reason: Custom Report Types allow users to generate insights that span multiple objects, such as Customers with their Loans or Loans with their Repayments.

Steps:

1. Go to Setup → Report Types → New Custom Report Type.
2. Customer with Loan Applications:
  - Primary Object: Customer\_\_c
  - Child Object: Loan\_Application\_\_c (Each ‘A’ record must have at least one related ‘B’ record)
  - Save & Deploy.



3. Loan with Repayments:
  - Primary Object: Loan\_Application\_\_c
  - Child Object: Repayment\_\_c
  - Save & Deploy.

Outcome: These report types enable building combined reports.

Customer with Loan Applications

Below is the information for this custom report type. You can click the buttons on this to preview or update information for the custom report type.

**Details**

- Display Label: Customer with Loan Applications
- API Name: Customer\_with\_Loan\_Applications
- Description: Customer with Loan Applications
- Created By: Shrindhi Shrinidhi, 9/24/25, 9:11 PM
- Store In Category: other
- Deployment Status: Deployed
- Modified By: Shrindhi Shrinidhi, 9/24/25, 9:11 PM

**Object Relationships**

Bank\_Customers (A) with at least one related record from Loan\_Applications (B)

A Venn diagram showing two overlapping circles labeled A and B. Below the diagram, there are two sets of horizontal bars, each labeled A and B, representing the fields included in the report.

## Step 2: Create Reports

What I did: I created reports using the custom report types for business insights.

Reason: Reports summarize data and provide detailed views for decision-making.

### 1. Loan Funnel:

- Report Type: Customer with Loan Applications
- Group by Loan\_Status\_\_c
- Columns: Customer Name, Loan Amount, Status
- Save as Loan Funnel Report.

FinSmart Banking A... Bank\_Customers ▾ Loan\_Applications ▾ Repayments ▾ Documents ▾ Offers ▾ \* Report Builder ▾ X

REPORT ▾

Loan Funnel Report Customer with Loan Applications

Outline Filters 2 Previewing a limited number of records. Run the report to see everything. Update Preview Automatically

	Customer Name	Loan_Application Name	Loan_Status
1	Priya Sharma	LA-0001	Draft
2	Sneha Reddy	LA-0002	Draft

Fields

Groups: GROUP ROWS Add group...

Columns: Add column...  Customer Name  Loan\_Application Name  Loan\_Status

## 2. Overdue EMIs:

- Report Type: Loan with Repayments
- Filter: Payment\_Status\_\_c = Overdue
- Columns: Loan Name, Customer, Due Date, Amount
- Save as Overdue EMIs Report.

The screenshot shows the Report Builder interface for FinSmart Banking A... The report is titled "Overdue EMIs Report" under the "Loan with Repayments" category. The report preview shows a table with the following data:

Payment_Status	Loan_Application_Name	Repayment_Name	Bank_Customer_Customer_Name	Due_Date	Due_Amount
Due (2)	LA-0001	RP-0001	Priya Sharma	10/23/2025	₹1,00,000.00
	LA-0002	RP-0002	Sneha Reddy	10/17/2025	₹50,000.00
					₹1,50,000.00
Total (2)					₹1,50,000.00

The report builder sidebar on the left includes sections for Outline, Filters (with one item selected), Groups (Group Rows and Payment\_Status), and Columns (Loan\_Application\_Name, Repayment\_Name, Bank\_Customer\_Customer\_Name, Due\_Date, # Due\_Amount). The bottom of the screen shows various report settings like Row Counts, Detail Rows, Subtotals, Grand Total, and Conditional Formatter.

## 3. Revenue by Loan Type:

- Report Type: Customer with Loan Applications
- Filter: Loan\_Status\_\_c = Disbursed
- Summarize Loan\_Amount\_\_c by Loan\_Type\_\_c
- Save as Revenue by Loan Type Report.

The screenshot shows the FinSmart Banking Application interface. At the top, there is a navigation bar with icons for search, report builder, and other system functions. Below the navigation bar, the main title is "FinSmart Banking A..." followed by dropdown menus for "Bank\_Customers", "Loan\_Applications", and "Repayments". A red box highlights the "Report Builder" tab. To the right of the tabs are buttons for "Save & Run", "Save", "Close", and "Run".

The main content area is titled "REPORT" and "Revenue by Loan Type". Below this, the specific report title is "Customer with Loan Applications". On the left, there is a sidebar titled "Fields" with sections for "Outline" (selected), "Filters" (with a count of 4), "Groups" (with "GROUP ROWS" selected), and "Columns" (listing "Customer Name", "Loan\_Application Name", and "Loan\_Status").

The main preview area shows a table with three columns: "Customer Name", "Loan\_Application Name", and "Loan\_Status". There are two rows of data:

	Customer Name	Loan_Application Name	Loan_Status
1	Priya Sharma	LA-0001	Draft
2	Sneha Reddy	LA-0002	Draft

A message at the top of the preview area says "Previewing a limited number of records. Run the report to see everything." and "Update Preview Automatically" with a checked checkbox.

### Step 3: Create Dashboard

What I did: I created a dashboard to visualize loan performance and repayment trends.

Reason: Dashboards provide management with quick access to KPIs and visual trends for better decision-making.

Steps:

1. Go to Dashboards → New Dashboard.
2. Name: Loan Management Dashboard, choose a folder.
3. Add Components:
  - KPI Tiles:
    - Total Disbursed Loans
    - Overdue Count
    - Average Credit Score
  - Charts:
    - Loans by Type → Pie/Bar chart
    - Overdue Trend → Line chart
4. Configure components to use reports from Step 2.
5. Save & Refresh Dashboard.

### Add Widget

**Report**

Loan Funnel Report (X)

Use chart settings from report ?

**Display As**

**Groups**

Add group... (X) Search...

**Columns**

Add column... (X) Search...

Customer Name (X)

Cancel Add

**Preview**

**Loan Funnel Report**

Customer Name ↑	Loan_Application Name	Loan_Status
Priya Sharma	LA-0001	Draft
Sneha Reddy	LA-0002	Draft

[View Report \(Loan Funnel Report\)](#)

FinSmart Banking A... Bank\_Customers Loan\_Applications Repayments \* Loan Management Dashb... X \* More ? Refresh Edit Subscribe

**Dashboard**  
**Loan Management Dashboard**  
As of Sep 24, 2025, 9:07 AM Viewing as Shrinidhi Shrinidhi

**Loan Funnel Report**

Customer Nam...	Loan_Application N...	Loan_St...
Priya Sharma	LA-0001	Draft
Sneha Reddy	LA-0002	Draft

[View Report \(Loan Funnel ... As of Sep 24, 2025, 9:07 AM](#)

**Overdue EMIs Report**

Sum of Due\_Amount  
Payment\_Status  
Due

₹150K

[View Report \(Overdue EMI... As of Sep 24, 2025, 9:07 AM](#)

## **Outcome**

- Reports summarize business data and provide insights.
- Dashboards visualize key loan and repayment metrics for managers.
- Sensitive fields are protected and security issues reviewed, ensuring data privacy.

# Phase 10 : Quality Assurance Testing

## Objective

The goal of this phase is to validate every feature built in FinSmart CRM (Phases 1–9) by preparing and executing test cases. Testing ensures the system meets business requirements, functions correctly, and is user-friendly.

## Test Cases

### Test Case 1 — Loan Application Record Creation

Use Case / Scenario: Loan Application Record Creation

Test Steps (Input):

- Navigate to Loan Applications tab.
- Create new Loan Application with Loan Amount = 600,000.

Expected Result: Loan\_Status\_\_c should auto-set to Under\_Review (trigger logic).

Actual Result: Loan saved successfully, Loan\_Status\_\_c = Under\_Review.

The screenshot shows the FinSmart CRM interface with the 'Loan\_Applications' tab selected. A modal window titled 'Information' is open, prompting for loan details. The fields filled are:

- \* Bank\_Customer: Shrinidhi
- \* Loan\_Type: Education
- \* Loan\_Amount: ₹6,000,000.00
- Interest\_Rate: (empty)
- Tenure\_Months: (empty)
- EMI: (empty)
- \* Loan\_Status: Draft
- Credit\_Score: (empty)

At the bottom of the modal, there are buttons for 'Assigned To', 'Cancel', 'Save & New', and a prominent blue 'Save' button.

The screenshot shows a mobile application interface for a loan application. At the top, there is a header with a bank icon and the text "Loan\_Application" and "LA-0010". Below the header, there are two tabs: "Related" and "Details", with "Details" being the active tab. The main area displays various loan application details in a table format:

LoanId	LA-0010
Bank_Customer	<u>Shrinidhi</u>
Loan_Type	Education
Loan_Amount	₹60,00,000.00
Interest_Rate	
Tenure_Months	
EMI	
Loan_Status	Under_Review
Credit_Score	

Each data entry field has a small edit icon (pencil) to its right.

### Test Case 2 — Validation Rule on Loan Amount

Use Case / Scenario: Validation Rule on Loan Amount

Test Steps (Input):

- Create new Loan Application with Loan Amount = 0.

Expected Result: Error message → 'Loan amount must be greater than 0.'

Actual Result: Validation rule fired correctly. (Screenshot placeholder)

**New Loan\_Application: Home Loan**

\* = Required Information

**Information**

LoanId

\* Bank\_Customer  
 ×

\* Loan\_Type  
 ▼

\* Loan\_Amount  
 ✖

Loan amount must be greater than 0

Interest\_Rate

Tenure\_Months

**We hit a snag.**

Review the following fields

- [Loan\\_Amount](#)

\* Loan\_Status  
 ✖

Cancel Save

### Test Case 3 — Approval Process for High-Value Loan

Use Case / Scenario: Approval Process for High-Value Loan

Test Steps (Input):

- Create Loan Application with Loan Amount = 1,200,000.
- Submit for Approval.

Expected Result: Loan\_Status\_\_c → Under\_Review. Approval request sent to Manager.

Actual Result: Approval request received. Loan\_Status\_\_c updated to Approved.

### Test Case 4 — EMI Reminder Scheduled Flow

Use Case / Scenario: EMI Reminder Scheduled Flow

Test Steps (Input):

- Create Repayment record with Due\_Date = Today.
- Run scheduled flow.

Expected Result: Customer receives reminder email.

Actual Result: Test email triggered successfully.

### **Test Case 5 — EMI Calculation (Apex Helper Class) Use**

Case / Scenario: EMI Calculation (Apex Helper Class)

Test Steps (Input):

- Run LoanHelper.calculateEMI(100000, 12, 12).

Expected Result: EMI > 0.

Actual Result: EMI calculated successfully = 8,885.

### **Test Case 6 — Credit Bureau Integration**

Use Case / Scenario: Credit Bureau Integration

Test Steps (Input):

- Insert Loan Application with status Submitted.
- Run CreditBureauService.getCreditScore(loan.Id).

Expected Result: Credit\_Score\_\_c updated (>=650 → Under\_Review, else Rejected).

Actual Result: Mock API returned 720. Loan updated to Under\_Review.

### **Test Case 7 — Data Import & Duplicate Rule Use**

Case / Scenario: Data Import & Duplicate Rule

Test Steps (Input):

- Use Data Import Wizard to import Customer.csv (duplicate Email).

Expected Result: Duplicate Rule blocks insertion and shows warning.

Actual Result: Duplicate error triggered correctly.

### **Test Case 8 — Reports Use**

Case / Scenario: Reports

Test Steps (Input):

- Navigate to Reports → Loan Funnel Report.

Expected Result: Loans grouped by Loan\_Status\_\_c displayed.

Actual Result: Report generated successfully.

### **Test Case 9 — Dashboard Use**

Case / Scenario: Dashboard

Test Steps (Input):

- Navigate to Dashboards → Loan Management Dashboard.

Expected Result: KPI Tiles and charts (Loans by Type, Overdue Trend, etc.) should appear.

Actual Result: Dashboard displayed as designed.

## **Conclusion**

The FinSmart CRM Project has successfully implemented Salesforce features covering data modeling, workflows, validation, automation, integration, reporting, and dashboards.

Final Outcome: FinSmart CRM provides a complete Loan & Customer Management solution with automation, compliance, real-time reporting, and seamless user experience for Bank Officers, Risk Analysts, Managers, and Customers.