Project Title: Finance Management System – Salesforce CRM

Phase 4: Process Automation (Admin)

Validation Rules :

Contribution

1. **Amount must be greater than zero**

Purpose
Ensure monetary integrity: contributions cannot be zero or negative.

Formula

Amount__c <= 0

Error Message
"Contribution amount must be greater than zero."

Impact
Prevents saving invalid financial records. Avoids negative/zero contribution values that would break roll-ups and financial reports.

Best Practices

- Create this rule early (before data imports).
- If you import historical data, temporarily disable rule during import and validate afterwards.
- Test with positive, zero and negative values.

2. **Payment Date cannot be in the future**

Purpose
Prevent entering future payment dates that would make transaction timing incorrect.

Formula

Payment_Date__c > TODAY()

Error Message
"Payment Date cannot be in the future."

Impact
Ensures chronological integrity of payments; important for reconciliation and reporting.

Best Practices

- If you allow scheduled future payments, use a separate field (Scheduled_Payment_Date__c) and do not apply this rule to that field.
- Test edge cases (today and timezone considerations).
- For DateTime fields use NOW() if appropriate.

3. Transaction ID required when Status = Paid

Purpose

Ensure auditability: when a contribution is marked Paid, record the external transaction reference.

Formula

AND(ISPICKVAL(Status__c, "Paid"), ISBLANK(Transaction_ID__c))

Error Message

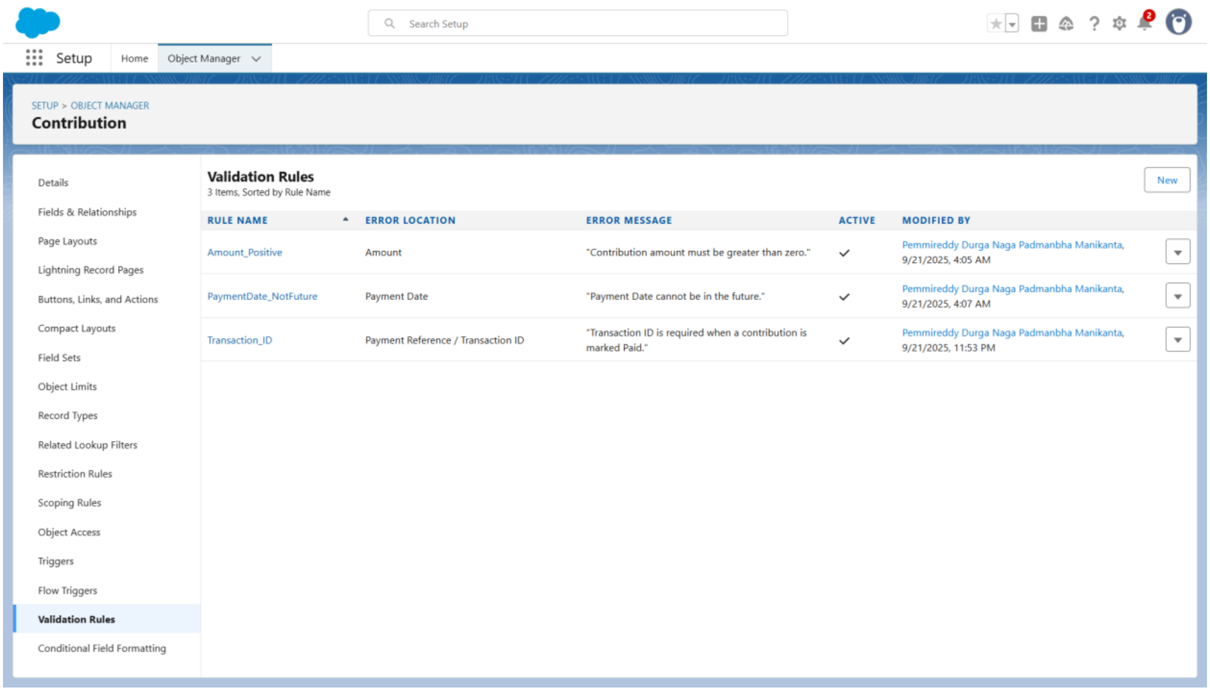
"Transaction ID is required when a contribution is marked Paid."

Impact

Improves traceability for finance/audit. Prevents marking payments as paid without proof/reference.

Best Practices

- Make Transaction_ID__c visible/editable for Agent and Manager profiles.
- Document the policy (cash vs electronic) in your SOP.



Auction

1. **Auction date cannot be in the past**

Purpose
Prevent scheduling auctions with dates earlier than today.

Formula

Auction_Date__c < TODAY()

Error Message
"Auction Date cannot be in the past."

Impact
Avoids incorrect scheduling and reporting errors; keeps auction calendar reliable.

Best Practices

- If auctions can be created historically for record-keeping, consider allowing a checkbox Is_Historical__c and exempt it from the rule with:
- Test with Today and Yesterday values.

2. **Prevent closing auction without Winner and Payout**

Purpose
Ensure auctions are only marked closed after a winner and valid payout are recorded.

Formula

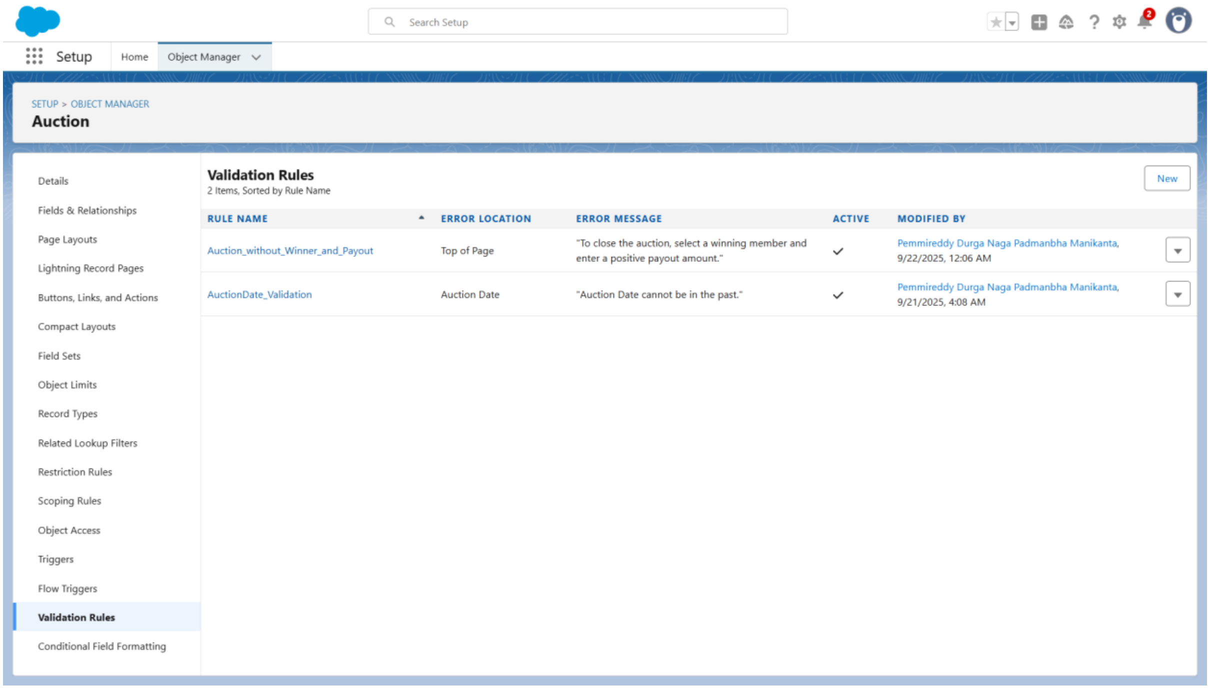
AND(Is_Closed__c = TRUE, OR(ISBLANK(Winner__c), Payout_Amount__c <= 0))

Error Message
"To close the auction, select a winning member and enter a positive payout amount."

Impact
Prevents incomplete/incorrect auction closure which would cause downstream accounting errors.

Best Practices

- Use this rule together with a flow that auto-sets Is_Closed__c after Business/Finance confirms payout.
- Keep the rule active only for profiles that close auctions (Manager/Admin), or set a bypass permission set if needed.



Member

1. KYC ID required for Premium Members

Purpose

Enforce compliance: premium members must have KYC documented.

Formula

AND(Is_Premium_Member__c = TRUE, ISBLANK(KYC_ID__c))

Error Message

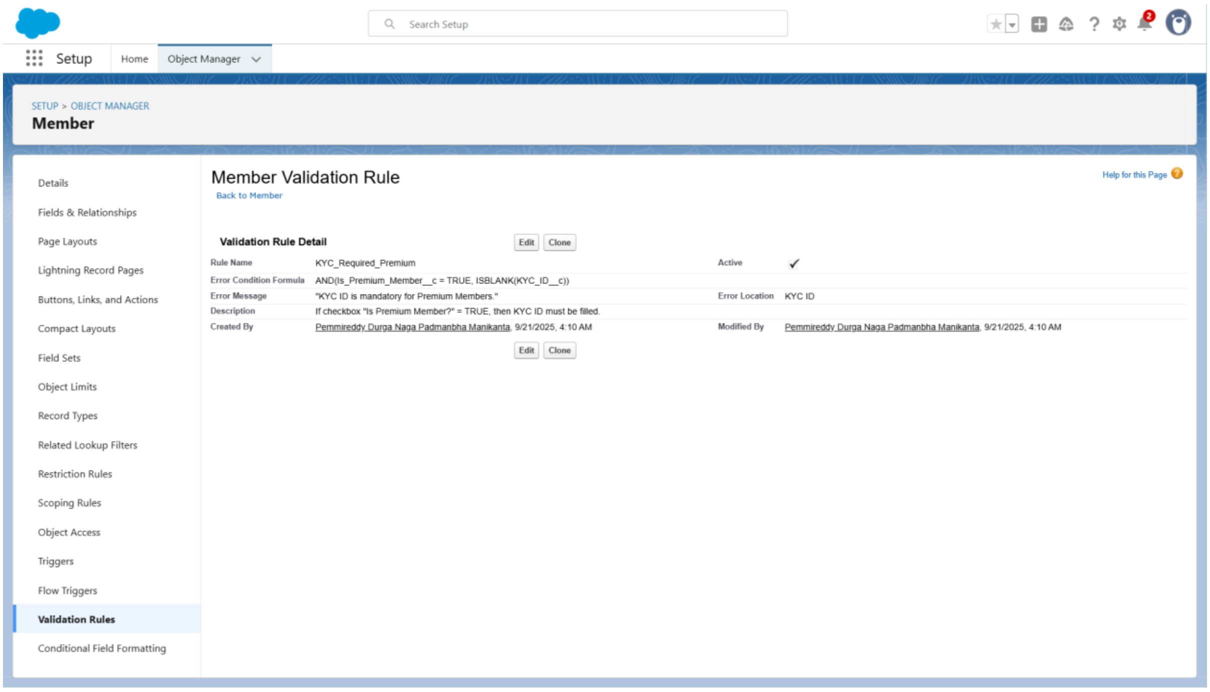
"KYC ID is mandatory for Premium Members."

Impact

Helps with regulatory compliance and reduces fraud risk for premium member privileges.

Best Practices

- Add Help Text on the KYC_ID__c field explaining acceptable KYC values.
- Make sure KYC_ID__c field-level security is restricted for sensitive profiles.
- If KYC is multiple document fields, consider checking all KYC-related fields OR use a checkbox KYC_Verified__c that admins toggle after manual verification, and validate that instead:
- AND(Is_Premium_Member__c = TRUE, NOT(KYC_Verified__c))



7) Notification: Date must be present

Purpose

Ensure every notification is time-bound (no floating/pending messages without a date).

Formula

ISBLANK(Date__c)

Error Message

"Notification Date must be specified."

Impact

Prevents creating notifications without scheduling info, which would break reminder automation and reporting.

Best Practices

- If you auto-populate Date__c in flows, consider making Date optional on layout but validate only when record is manually created: add condition ISNEW() if needed:
- Make Date a Date/Time if exact timestamp matters.

8) Notification: SMS requires Member phone

Purpose

Prevent attempts to send SMS notifications when the member has no phone number.

Formula

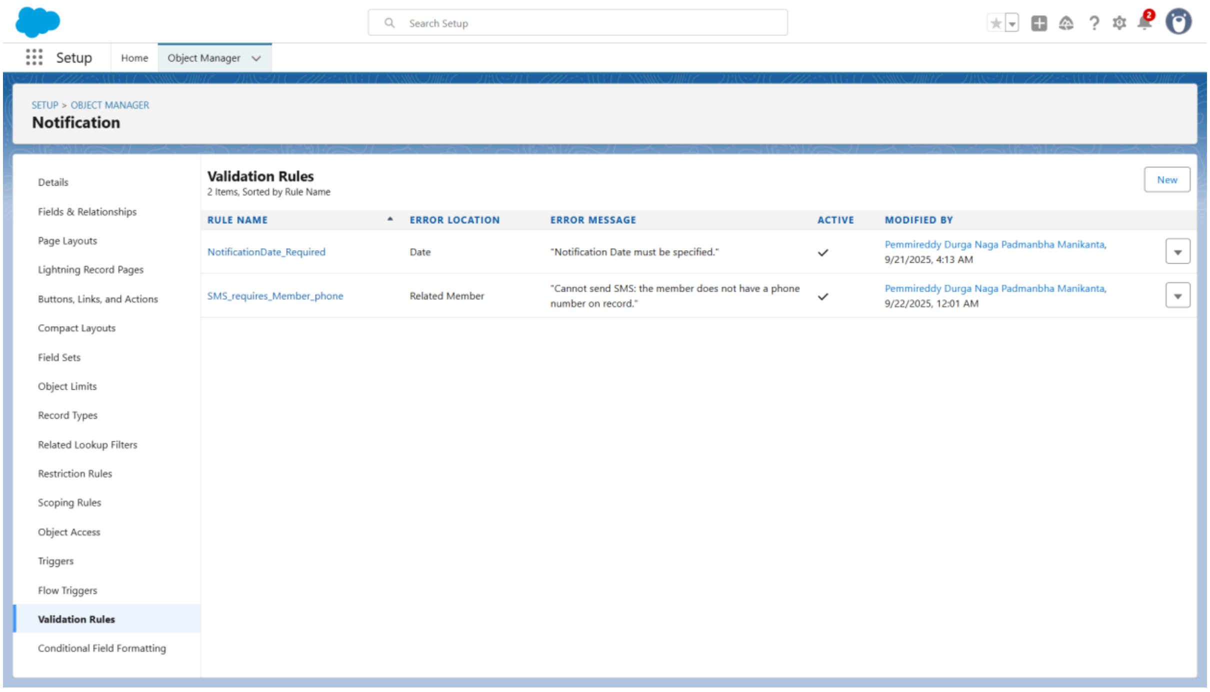
AND(ISPICKVAL(Notification_Type__c, "SMS"), ISBLANK(Related_Member__r.Phone))

Error Message
"Cannot send SMS: the member does not have a phone number on record."

Impact
Avoids failed SMS sends and reduces wasted callouts/credits. Ensures better automation quality.

Best Practices

- If some members store phone on Contact rather than Account, adapt the reference accordingly (e.g., Related_Member__r.Contact_Phone__c).
- Provide a UI helper (Flow/Screen) to capture a phone if missing.



WorkFlow Rules :

<u>Contribution</u>

1. Paid → Send Payment Confirmation Email

Purpose
Notify members instantly when their contribution is recorded as Paid, building trust and reducing manual follow-ups.

Formula (Rule Criteria)

ISPICKVAL(Status__c, "Paid")

Error Message
N/A – Workflow Rules do not block saves.

Impact

- Improves transparency.
- Gives members proof of payment.
- Reduces disputes about contributions.

Best Practices

- Use an **email template with merge fields** (Member Name, Amount, Payment Date).
- Test the rule with multiple contribution records before activating.
- Use **Flows for future replacements**, since Workflow Rules are legacy.

2. Overdue → Notify Manager

Purpose
Alert Finance Managers when a contribution is overdue so they can take action quickly.

Formula (Rule Criteria)

ISPICKVAL(Status__c, "Overdue")

Error Message
N/A

Impact

- Ensures managers are aware of overdue contributions.
- Improves collection efficiency.
- Prevents revenue leakage.

Best Practices

- Send to a **distribution list or queue** if multiple managers are responsible.
- Combine with **Flows** for creating tasks or notifications.
- Document escalation steps for overdue follow-ups.

Member

1. Send Welcome Email

Purpose
Send a welcome message whenever a new Member record is created. Builds goodwill and introduces members to the system.

Formula (Rule Criteria)

TRUE

Error Message
N/A

Impact

- Enhances onboarding experience.
- Creates a transparent communication channel from the start.
- Reinforces professionalism of the chit fund/finance company.

Best Practices

- Use a **professional branded email template**.
- Keep content clear (welcome note, next payment details, contact info).
- Consider using **Flows** if you want to schedule the welcome email after X days.

Auction

1. Closed → Notify Winner

Purpose
When an auction is closed, notify the winning member instantly.

Formula (Rule Criteria)

Is_Closed__c = TRUE

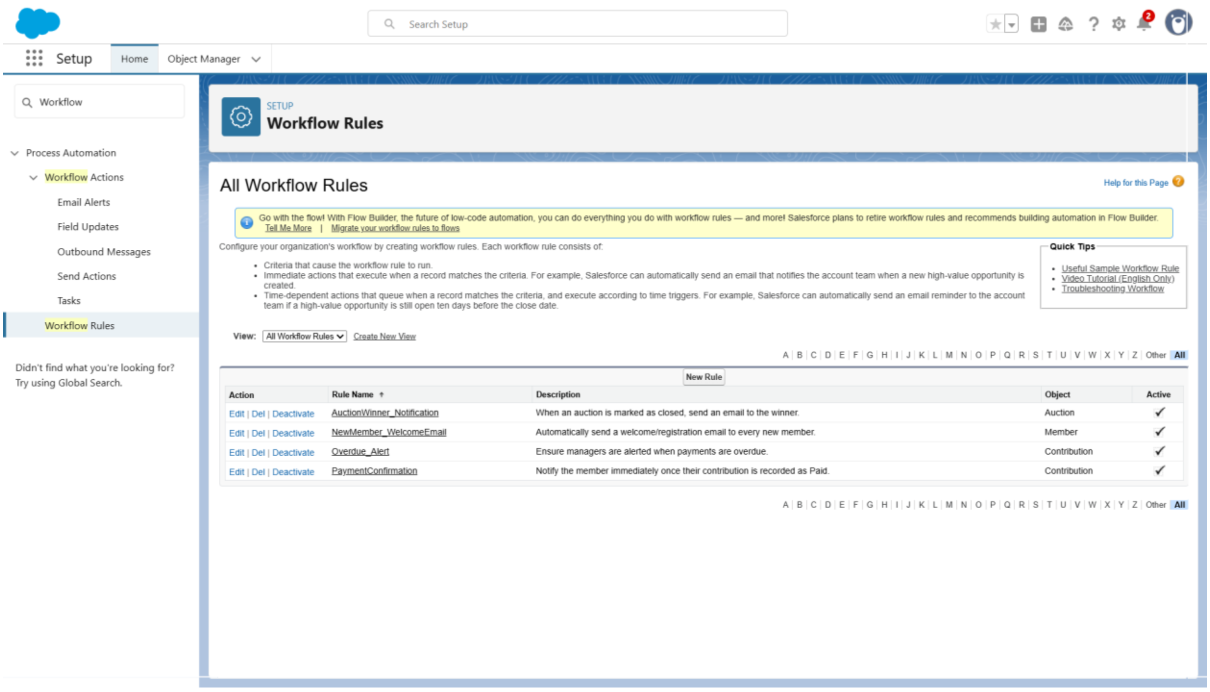
Error Message
N/A

Impact

- Ensures winner receives timely confirmation.
- Prevents confusion about auction results.
- Keeps auction process transparent and fair.

Best Practices

- Ensure **Winner__c lookup** is filled before closing auction (else email fails).
- Include **auction details** in the email (date, bid amount, payout).
- Test with sample auctions before enabling for all users.



Process Builder

Auto-update Contribution Status → Paid

Purpose

Automatically update the **Contribution Status** to "Paid" when the **Payment Date** field is entered. This reduces manual updates and ensures data accuracy.

Formula (Criteria for Executing Actions)

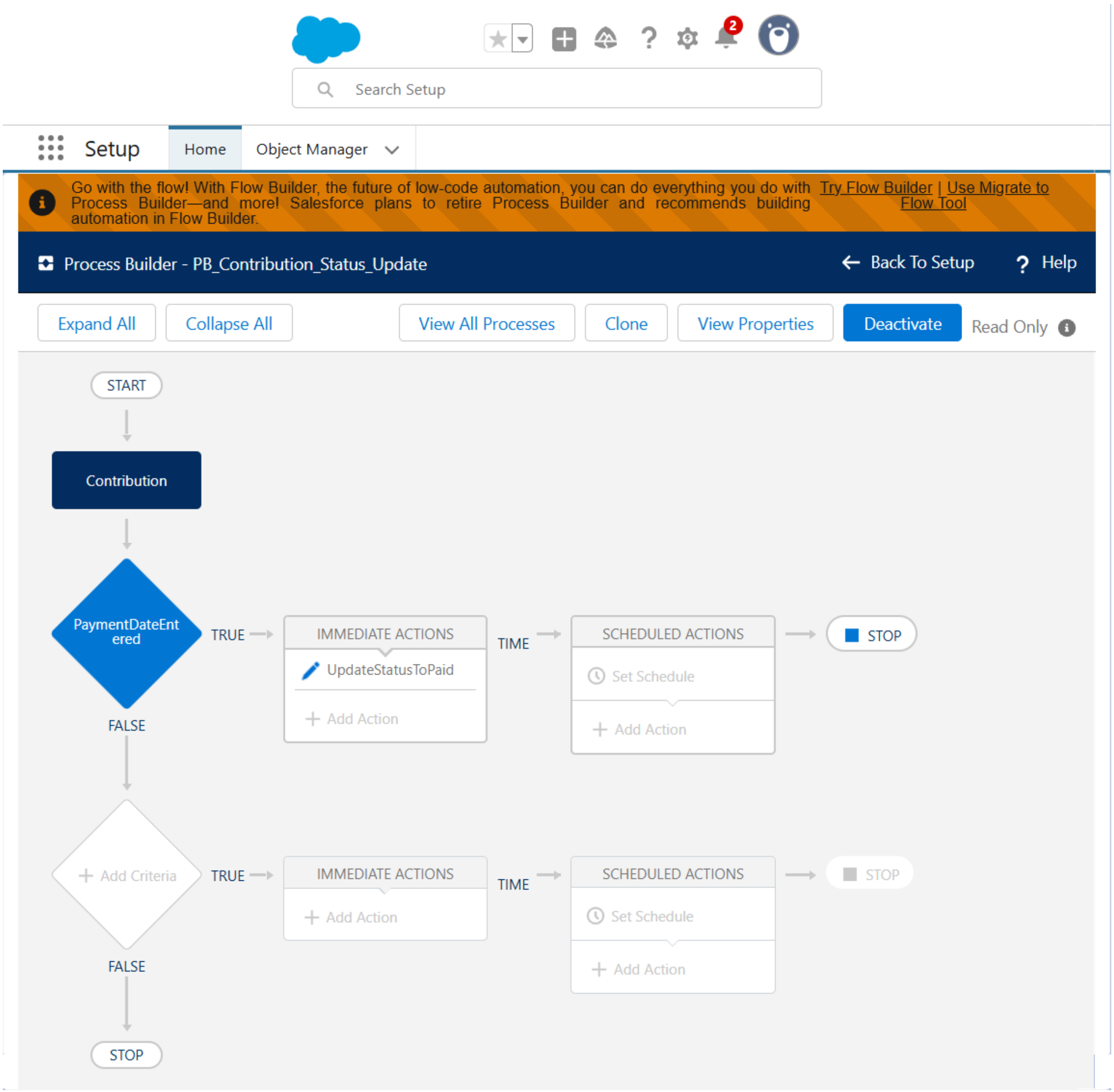
NOT(ISBLANK([Contribution__c].Payment_Date__c))

Impact

- Saves agents time by auto-updating records.
- Ensures that Payment Date and Status fields stay consistent.
- Reduces errors where Payment Date is filled but Status is left as Pending.
- Improves accuracy of reports and dashboards that rely on Contribution Status.

Best Practices

- Use **Flows** for new automations, as Process Builder is legacy and will eventually be retired.
- Add a **Validation Rule** to prevent Status = Paid if Payment Date is blank (to complement this automation).
- Clearly **name the process** (PB_Contribution_Status_Update) to make it easy to identify in Setup.
- Always **test with sample records** before activating for all users.



Approval Process

1. **Objective**
   To control contributions exceeding ₹50,000 by requiring Finance Manager approval, ensuring compliance with organizational policies and reducing financial risk.

2. **Entry Criteria**
   Amount__c > 50000

3. **Initial Submitters**

- Contribution Owner (Agent)
- Managers
- Finance Admins

4. **Initial Submission Action**
   a. Record is **locked** to prevent edits during review.
   b. Contribution.Status__c updated to **"Pending Approval."**
   c. Email notification sent to assigned approver (Finance Manager).

5. **Approval Steps**
   a. **Step 1:** Finance Manager approval is required.
   b. If approved → moves to Final Approval Actions.
   c. If rejected → moves to Final Rejection Actions.

6. **Final Approval Actions**
   a. Contribution.Status__c updated to **"Approved."**
   b. Record can remain locked to preserve integrity (or unlocked if configured).
   c. Email notification sent to Contribution Owner and Member confirming approval.

7. **Final Rejection Actions**
   a. Contribution.Status__c updated to **"Rejected."**
   b. Record unlocked so the Agent can make corrections or resubmit.
   c. Email notification sent to Contribution Owner explaining rejection.

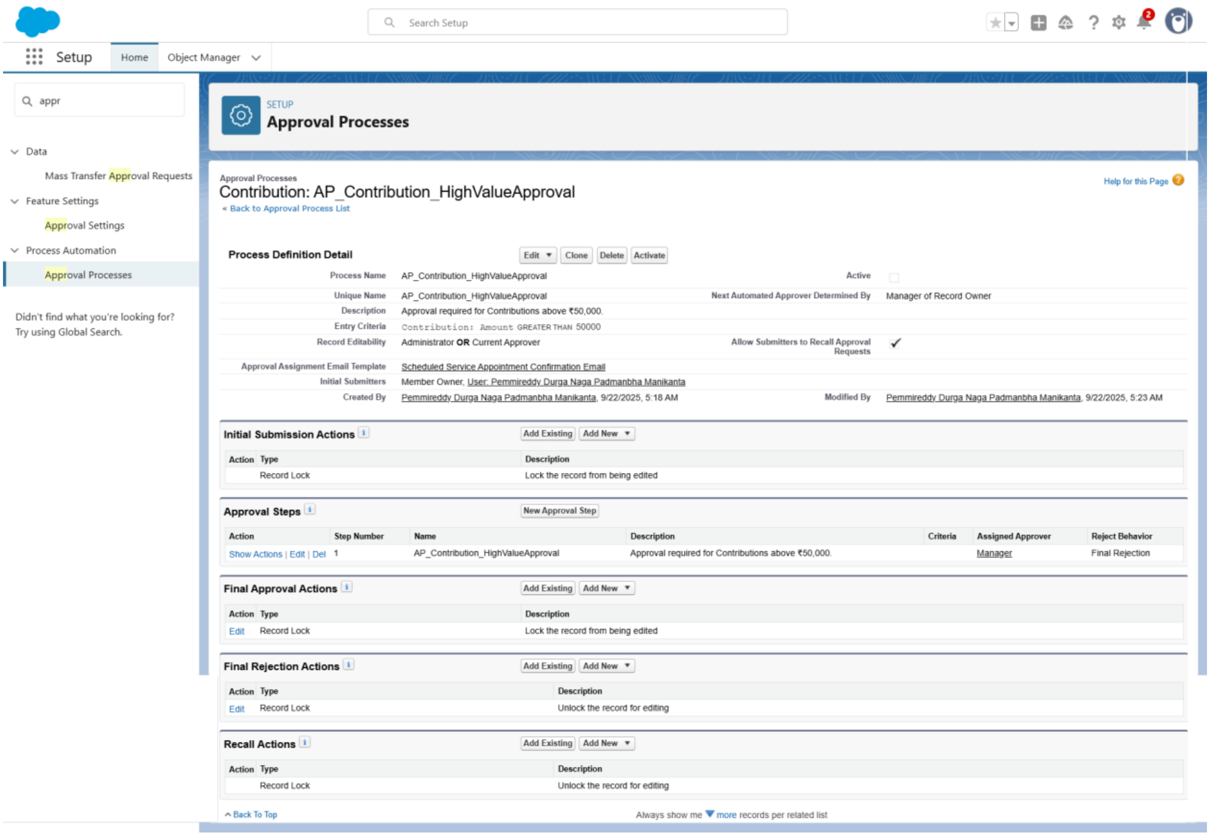
8. **Recall Actions**
   a. If submitter recalls, record is **unlocked** and editable again.

9. **Impact**

- Prevents unauthorized approval of high-value contributions.
- Ensures Finance Managers review sensitive transactions.
- Adds audit tracking through Approval History.
- Improves compliance, accountability, and member trust.

Example Test

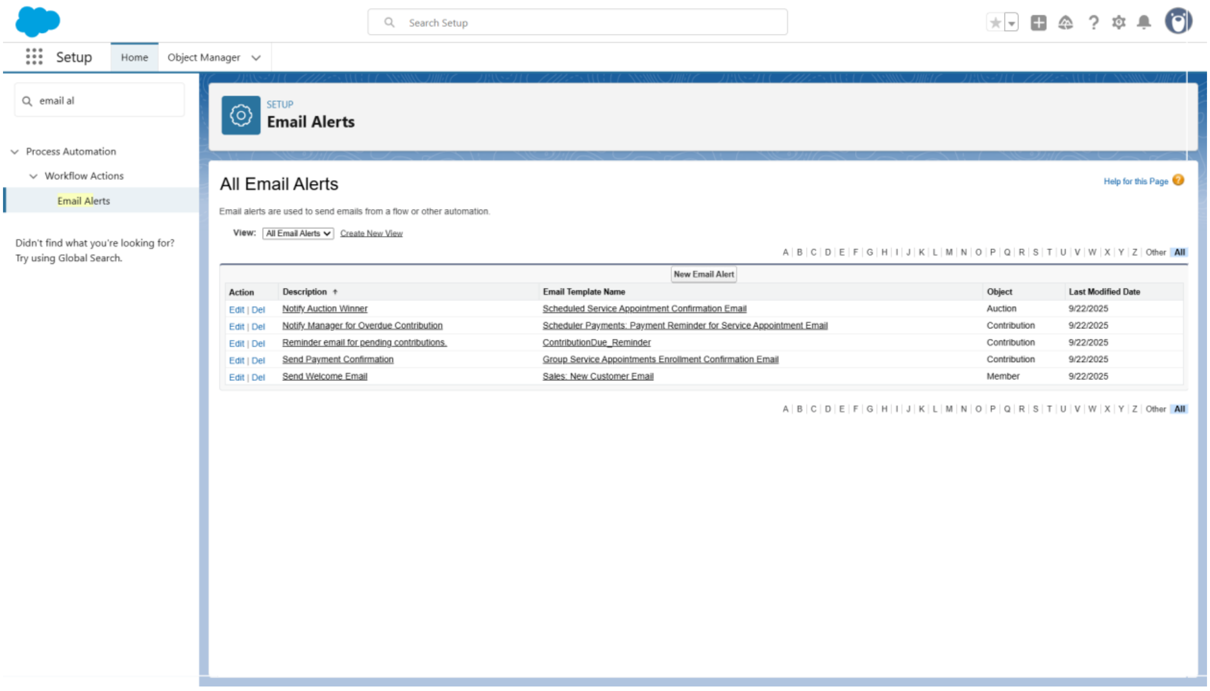
1. Create Contribution with Amount = ₹60,000.
2. Save → Salesforce sends approval request to Finance Manager.
3. Approver receives email + sees request in Approval tab.
4. Approver **approves** → Contribution.Status__c updated to “Approved.”
5. Approver **rejects** → Contribution.Status__c updated to “Rejected.”



Flow builder

I have implemented Flow Builder to configure, Custom Notifications, and Task automation. The respective screenshots have been included for reference.

Email alerts



1. Email Alert Configuration

To automate contribution due reminders, an Email Alert was created in Salesforce. This reusable action links an email template to recipients and can be triggered by Flows or other automation tools.

- **Alert Name:** EA_ContributionDue_3d
- **Object:** Contribution__c
- **Email Template:** ContributionDue_Reminder_3d (Lightning Email Template)
- **Sender Address:** Org-Wide Email Address (ensures consistent communication)
- **Recipients:** Contribution Owner (Agent) and Related Member (via lookup field)
- **Purpose:** Ensures contribution due reminders are sent in a standardized, timely manner.

2. Integration with Record-Triggered Flow

A Record-Triggered Flow was built to send reminders three days before the Contribution Payment Date.

- **Trigger:** Contribution__c created or updated
- **Entry Conditions:**

- Payment_Date__c is not null
- Status__c = Pending
- Reminder_Sent__c = FALSE

- **Scheduled Path:** Executes 3 days before Payment_Date__c
- **Action:** Calls the Email Alert (EA_ContributionDue_3d), passing the Contribution record Id
- **Duplicate Prevention:** Reminder_Sent__c flag is updated to TRUE after sending

Result: Each pending contribution automatically generates a reminder exactly three days before its due date.

3. Integration with Scheduled-Triggered Flow

A Scheduled-Triggered Flow was also developed to handle bulk processing of reminders.

- **Schedule:** Runs daily at a fixed time (e.g., 9 AM)
- **Query Logic:** Retrieves all Contribution__c records where:
    - Payment_Date__c = Today + 3
    - Status__c = Pending
    - Reminder_Sent__c = FALSE
- **Loop Logic:** For each qualifying record:
    - Sends the Email Alert EA_ContributionDue_3d
    - Updates Reminder_Sent__c = TRUE to avoid duplicates
- **Bulk Update:** Records are updated in a collection for efficiency

Result: Complements the record-triggered flow by ensuring no contribution is missed, even if Payment Dates were updated historically.

<u>Field Updates</u>

Field Updates were not used because they are deprecated, have limited flexibility (can only set simple values), and are being replaced by Flows, which provide more advanced, scalable, and future-proof automation.

Tasks

To automate reminders for upcoming contribution due dates, I implemented a **Scheduled Flow** on the Contribution__c object. This flow automatically creates **Tasks** for contribution owners (agents) **3 days before the Payment Date** and ensures duplicate reminders are not sent.

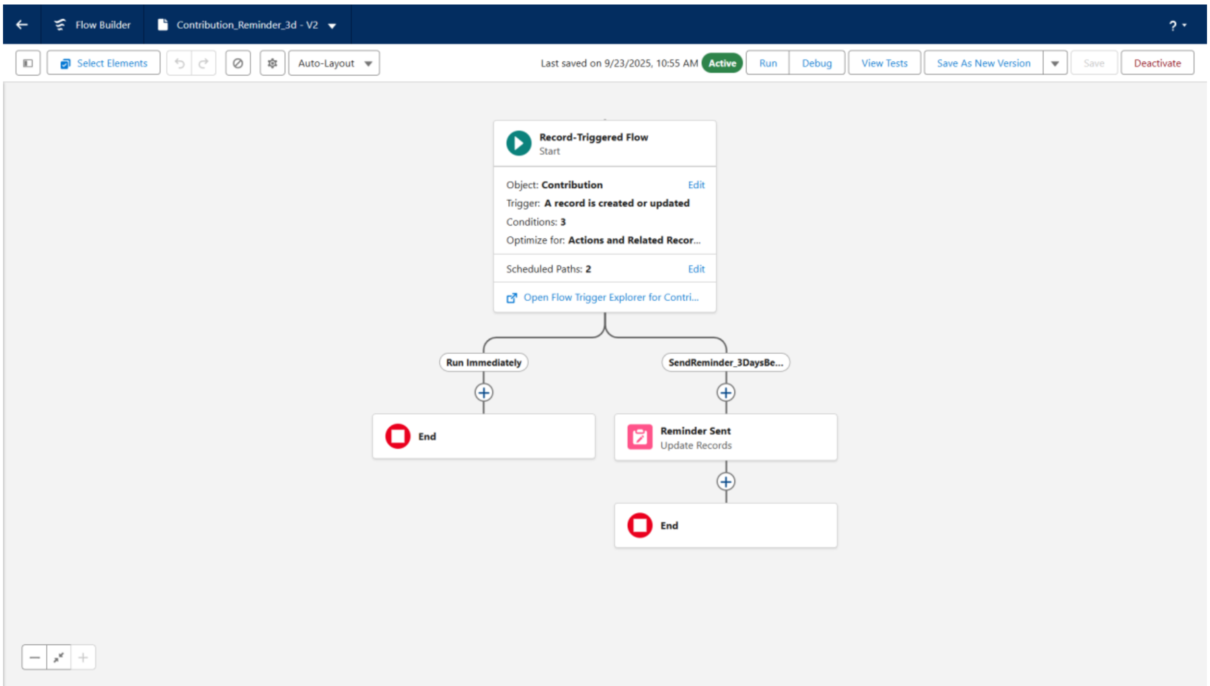
Flow Configuration Steps

1. **Create Scheduled Flow**
    - Setup → Flows → New Flow → **Scheduled-Triggered Flow**.
    - Frequency: **Daily** | Start Time: **2:00 AM**.
2. **Get Records**
    - Object: **Contribution__c**
    - Criteria:
        - Payment_Date__c = Today + 3
        - Status__c = Pending
        - Reminder_Sent__c = False
    - Store all matching records in a collection.
3. **Decision Element**
    - If no records found → End Flow.
    - If records found → Continue to loop.
4. **Loop Through Contributions**
    - Iterate through each Contribution record in the collection.
5. **Create Task (inside loop)**
    - **Subject:** Reminder: Follow up with member for contribution due.
    - **OwnerId:** Contribution Owner (Agent).
    - **WhatId:** Contribution record Id.
    - **WhoId:** Member__c (if linked).
    - **Due Date:** Today (3 days before due date).
    - **Status:** Not Started.
    - **Priority:** High.

    - **Description:** Includes Member Name, Payment Amount, Due Date, and Record Link.

6. **Update Contribution (inside loop)**
    - Update field Reminder_Sent__c = True to prevent duplicate tasks.

7. **End Loop & Activate Flow**
    - Save as ContributionDue_3d.
    - Activate flow.



Outcome

This flow ensures that contribution owners (agents) receive a **high-priority Task reminder exactly 3 days before the contribution is due**, improving collection management and reducing defaults.

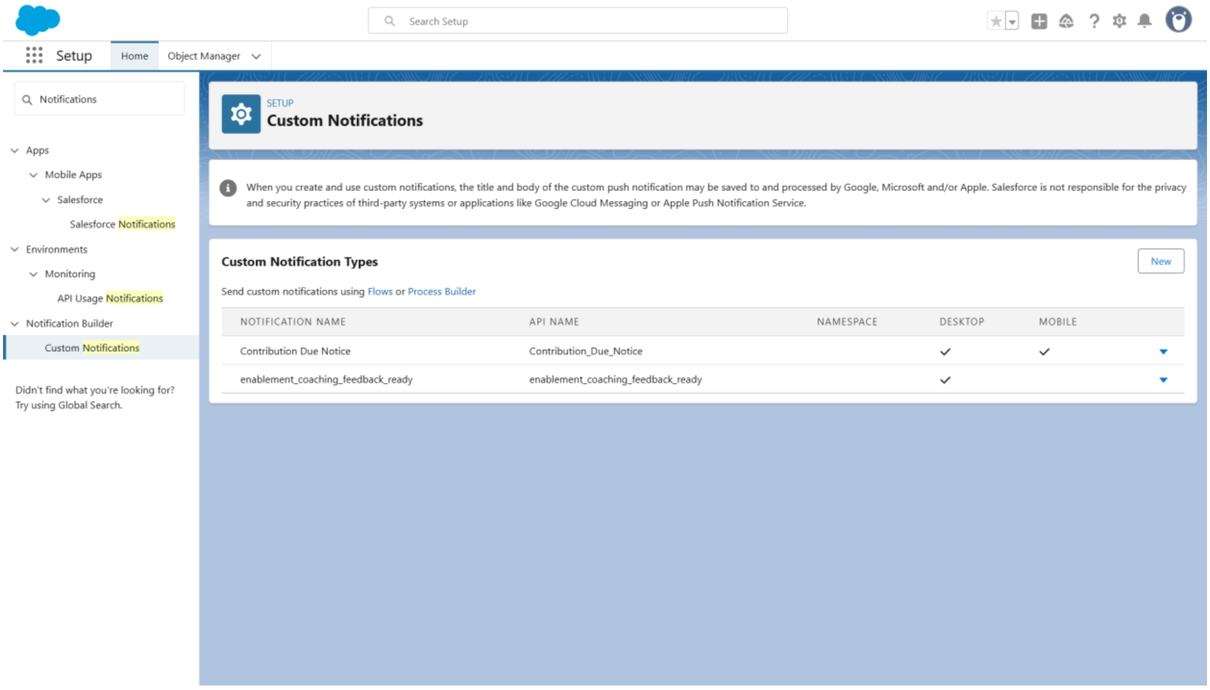
Custom Notifications

To enhance contribution and auction management, I created a **Custom Notification Type** and a **Scheduled Flow** to automatically notify contribution owners **3 days before a contribution's Payment Date**. This ensures proactive follow-up and improves efficiency.

Step 1: Create a Custom Notification Type

- Navigation: Setup → **Notification Builder** → **Notification Types** → **New**.
- Details:

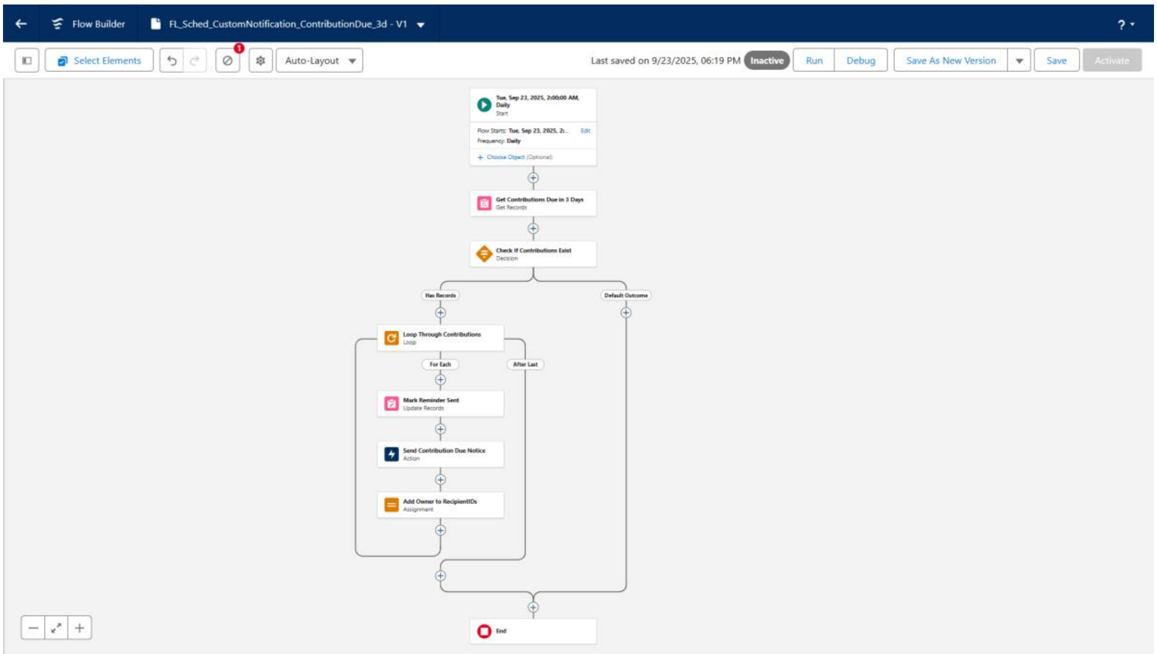
- **Label:** Contribution Due Notice
- **API Name:** Contribution_Due_Notice
- **Supported Channels:** Desktop, Mobile, and In-App (bell icon).
- **Description:** Notification for upcoming contribution dues.



Step 2: Build the Scheduled Flow

1. **Flow Setup**
   - Setup → Flows → New Flow → **Scheduled-Triggered Flow**.
   - Frequency: **Daily** | Start Time: **2:00 AM**.
2. **Get Contributions to Notify**
   - Object: **Contribution__c**
   - Criteria:
     - Payment_Date__c = Today + 3
     - Status__c = Pending
     - Reminder_Sent__c = False
   - Store all matching records in a collection.
3. **Decision Check**
   - If records exist → Continue.
   - If none → End flow.

4. **Loop Through Records**
   - Iterate through each Contribution in the collection.
5. **Recipient Collection Setup**
   - Create a **Text Collection Variable** (RecipientIDs).
   - Add Contribution Owner (OwnerId) into this collection.
6. **Send Custom Notification (inside loop)**
   - Notification Type: **Contribution Due Notice**.
   - **Title:** Contribution due in 3 days: {!Contribution__c.Name}
   - **Body:** Member {!Contribution__c.Member__c} owes {!Contribution__c.Amount__c} by {!Contribution__c.Payment_Date__c}.
   - **Target Id:** Contribution record Id.
   - **Recipient Ids:** RecipientIDs collection.
7. **Cleanup Recipients**
   - Remove the owner from RecipientIDs after each loop to avoid carry-over.
8. **Update Contribution**
   - Set Reminder_Sent__c = True for each processed contribution.
9. **Activate Flow**
   - Saved as FL_Sched_CustomNotification_ContributionDue_3d and activated.



Outcome

This automation sends a **real-time Salesforce in-app and mobile notification** to contribution owners 3 days before a payment is due, helping them take timely action. The system also updates each contribution to avoid duplicate notifications.