Flows Guidelines (Low Code)

General Guidelines

Modular Design & Architecture

- Single Responsibility Principle: Each flow should handle ONE specific business process or function
- Maximum Elements Guideline: Aim for flows with fewer than 50 elements; if exceeding, consider splitting
- Subflow Strategy: Extract reusable logic into subflows (e.g., validation logic, notification logic, calculation logic)
- Flows First: Use flows for all business logic. If flows cannot be used, a justification must be provided.

Flow Version Management

IMPORTANT: Pay attention to the Flow API version

- Build your flow using the same version available in Production at the Go Live Date
- Recommendation: Always use the current API Version -1
  - Example: If the current version available is 60, use version 59
- This ensures stability and avoids issues with newly released features that may have bugs

Flow Trigger Explorer Management

MANDATORY: Use Flow Trigger Explorer to explicitly set execution order for all Record-Triggered Flows on the same object

- When adding a record trigger flow, always reflect on the order of execution and setup a proper order compared to the other record trigger flows.

Naming Conventions

Flow API Name Pattern

Object-Specific Flows:	[Prefix]_[Object]_[Flow Description]_[Suffix]
Non-Object-Specific Flows:	[Prefix]_[Flow Description]_[Suffix]

The prefix represents the process category or business domain the flow belongs to.

Reference: See the Domain Prefix Mapping Table (Annex I) in the Appendix for the complete list of prefixes.

The suffix represents the type of event:

- Before Save: _BS
- After Save: _AS

- Before Delete: _BD

Examples:

- ACC_Account_ValidateAddress_BS - Account process flow for address validation
- OPP_Opportunity_CalculateDiscount_BS - Opportunity process flow for discount calculation
- GEN_SendEmailNotification_AS - Generic/cross-process subflow for sending email notifications

Email Notifications

MANDATORY: Use Email Alerts instead of single emails

- Email Alerts allow handling the email body with an Email Template instead of managing it manually in the flow
- This provides better maintainability and allows business users to update email content without modifying flows
- Exception: Single emails are acceptable only when reaching Email Alert limits

Error Handling & Logging

Default Error Handling

Uncaught exceptions in flows are automatically captured in the persistent Log Message table via platform events fired by Salesforce. This provides baseline error tracking without additional configuration.

Fault Path Implementation (When Required)

SELECTIVE USE: Fault paths should NOT be added systematically to every element. Only implement fault paths when a business exception is expected and requires special behavior.

When to Add Fault Paths:

Scenario	Reason for Fault Path
Business validation failures	Need to display user-friendly message or redirect flow
Integration callout failures	Need to implement retry logic or fallback behavior
Conditional rollback requirements	Need to partially commit or handle specific DML failures
User notification requirements	Need to alert specific users or teams about the failure
Custom recovery logic	Need to execute alternative business process on failure

When NOT to Add Fault Paths:

- Standard DML operations where default error handling is sufficient
- Simple field updates where failure should stop the process
- Operations where the default platform event logging provides adequate visibility

Performance & Bulkification

Loop Optimization

NEVER place DML operations (Create, Update, Delete) inside loops
NEVER place Get Records inside loops

Correct Pattern:

1. Collect records in a collection variable within the loop
2. Perform single DML operation outside the loop

Governor Limit Awareness

Limit Type	Maximum per Transaction
SOQL Queries	100
DML Statements	150
Records Retrieved	50,000
Flow Interviews	2,000

Entry Conditions

MANDATORY: Define precise entry conditions to prevent unnecessary flow executions

- Use $Record__Prior comparisons to detect actual field changes
- Example: Only run when Status changes from any value to "Closed"

Record-Triggered Flows - Synchronous & Asynchronous Paths

Bypass Automation Check

MANDATORY: Flows at the beginning of execution should verify if they are bypassed

- Check a ByPassAutomation custom permission at the start of the flow
- This allows administrators to temporarily disable automations for data migrations or troubleshooting
- Implement as the first Decision element in the flow

Flows First Approach

- Use flows for all business logic
- If flows cannot be used, a justification must be provided
- Wherever Apex is required for lightweight operations, it should be called from the trigger flow
    - Examples: Apex sharing, formula recalculation, complex calculations

Asynchronous Path Limitations

IMPORTANT: Only ONE asynchronous flow per object

- Since we cannot force the order of execution for Asynchronous Flows, we should have only one async flow per object
- This prevents unpredictable behavior and race conditions
- If multiple async operations are needed, consolidate them into a single async path with decision logic

Test Coverage Requirements

Synchronous Path

- Test classes are mandatory for synchronous paths
- Build during the build phase following normal Apex testing guidelines
- Include assertions to validate expected outcomes

Asynchronous Path

KNOWN LIMITATION: Due to current Salesforce limitations, the Test.stopTest() method does not function as expected for asynchronous paths of autolaunched flows.

Approach:

- We are unable to validate the results of asynchronous path flows in unit tests
- This is different from testing @future methods and Queueables where results can be validated after Test.stopTest()
- Exception: We do not include assertions in unit tests for asynchronous paths of autolaunched flows
- These unit tests are created solely to achieve code coverage, enabling the deployment of flows as active

Future Benefit: A positive outcome of this approach is that we will have pre-existing unit tests ready for when Salesforce introduces the capability to verify asynchronous flow results in tests.

Record-Triggered Flows - Scheduled Paths

Lead Object Restrictions

IMPORTANT: Avoid scheduled paths on the Lead object unless the time offset is very short (a day or less)

Reason: There is a known issue where a Lead cannot be converted if it has pending time-based actions.

Workaround: If an action needs to happen X days after a trigger:

1. Create a Scheduled Flow running daily
2. Check if the criteria is met (e.g., if 30 days passed after creation)
3. Execute the required actions in that scheduled flow

General Scheduled Path Guidelines

- Keep time offsets as short as practical
- Consider the impact on record operations during the scheduled window
- Document the expected timing and business justification for scheduled paths

Screen Flows

Screen Flows as Preferred Solution

RECOMMENDED: Screen flows and dynamic layouts are the recommended solution over LWC

Assessment Process:

1. Every new requirement (User Story) should first be assessed to determine if it can be achieved using a no-code solution
2. If requirements cannot be achieved via screen flow, they should be challenged to assess and provide a simplified proposal using the standard no-coding approach
3. Only after exhausting no-code options should custom development be considered

LWC Embedding

- It is acceptable to include LWC components into screen flows
- Balance is key: Maintain the right balance between flow elements and custom components
- Use LWC only for functionality that cannot be achieved with standard flow components

Test Coverage

NOTE: Screen flows do not require Apex test coverage to be deployed

- It is not possible to write Apex tests for screen flows
- Testing should be performed manually through the UI
- Document test scenarios for QA validation

Flow Test Coverage Summary

When Test Coverage is Required

Flow Type	Test Coverage Required	Notes
Record-Triggered (Synchronous)	Yes - Mandatory	Full assertions required to deploy as active
Record-Triggered (Asynchronous)	Yes - Coverage only	No assertions due to platform limitations
Record-Triggered (Scheduled Path)	Yes - Coverage only	Similar limitations as async
Screen Flow	No	Cannot be tested via Apex
Scheduled Flow (Schedule-Triggered)	No	Not required by Salesforce
Platform Event-Triggered	No	Not required by Salesforce
Autolaunched (Subflow)	Depends	Covered when parent flow is tested

Note: Salesforce only requires test coverage for Record-Triggered Flows to deploy them as active. Other flow types (Scheduled Flows, Platform Event-Triggered Flows, Screen Flows) do not require Apex test coverage for deployment.

Test Coverage Best Practices

- Positive: Testing: Test happy flow scenario
- Negative Testing: Test scenarios where entry conditions are NOT met (flow should not execute)
- Edge Cases: Test null values, empty collections, and boundary conditions
- In Syensqo all flows are deployed as active therefore the flow coverage of the org (number of flows that are ‘called’ during apex tests execution divided by number of all active flows) should be greater than 75%. (see Queries to calculate flow coverage to have details on how to calculate flow coverage)
- In relation to the Apex tests, point and click flow tests are not considered part of test coverage. It lead to the situation that developers needs to assure that AutolaunchedFlow, CustomEvent flows, invocable flows and scheduled are covered (executed at least once) by apex tests

Important: Apex Tests vs Flow Tests

CRITICAL DISTINCTION: Salesforce offers two types of flow testing, but only one counts for deployment:

Test Type	Counts for Coverage	Runs During Deployment
Apex Tests (test classes that trigger flows)	Yes	Yes
Flow Tests (point-and-click tests in Flow Builder)	No	No

Flow Tests created in Flow Builder are useful for manual validation and debugging, but they:

- Are NOT executed during deployment
- Do NOT count toward flow coverage requirements

Therefore, Apex test classes are required to achieve the coverage needed for deploying Record-Triggered Flows as active.

Documentation Requirements

Flow Description

Every flow MUST include a clear description explaining what the flow does:

This flow validates the billing address format for Account records when they are created or updated.

It checks for required address fields and standardizes the format before saving.

Keep descriptions:

- Concise: Focus on what the flow does, not implementation details
- Business-focused: Explain the business purpose
- Up-to-date: Update when flow functionality changes

Element Descriptions

- Add descriptions to complex Decision elements explaining the business logic
- Document any "magic numbers" or hardcoded values with business justification

Anti-Patterns to Avoid

DO NOT:

- Create multiple Record-Triggered Flows on same object without using Flow Trigger Explorer
- Place SOQL or DML inside loops
- Use hardcoded IDs
- Add fault paths to every element systematically (only when business exception handling is needed)
- Leave flows without descriptions
- Create overly complex single flows (50+ elements)
- Use Process Builder or Workflow Rules (migrate to Flow)
- Use single emails instead of Email Alerts (unless hitting limits)
- Create multiple asynchronous flows on the same object
- Use scheduled paths on Lead object with long time offsets
- Build custom LWC when screen flows can achieve the requirement

Workflow history

This view shows the 5 most recent entries. The complete workflow log is available from the 'Document Activity' menu item.

From Jan 28, 2026 to Jan 30, 2026	Actor	Type	Activity	Version
	KAROLINSKI-ext, Stephane and BROWAEYS-ext, David	Edit	multiple updates from KAROLINSKI-ext, Stephane and BROWAEYS-ext, David	
	BROWAEYS-ext, David	Edit	created the page at 2:32 pm	