

CLIENTELL TECHNICAL DEBT SERIES

# Fixing Salesforce Technical Debt With AI

Worksheets, AI prompts, and a 90-day plan to audit, prioritize, and fix your org's accumulated shortcuts. For admins and the leadership teams who fund them.

**67%**

Of admins say tech debt is their #1 daily challenge

**200+**

Custom fields per object (average org)

**30<sup>K</sup>**

Stale reports in the average enterprise org

**12%**

Revenue lost to bad CRM data (Gartner)



SOC 2



HIPAA



GDPR



SALESFORCE PARTNER

**Neil Sarkar**

CTO &amp; CO-FOUNDER, CLIENTELL

CLIENTELL.COM

© 2026 CLIENTELL, INC.

# The Weight You're Carrying

67% of Salesforce admins say technical debt is their #1 daily challenge. Not reports. Not user requests. Not even Flows. Technical debt — the invisible, compounding mess that makes every "simple change" take three times longer than it should.

WHAT THE AVERAGE SALESFORCE ORG LOOKS LIKE RIGHT NOW

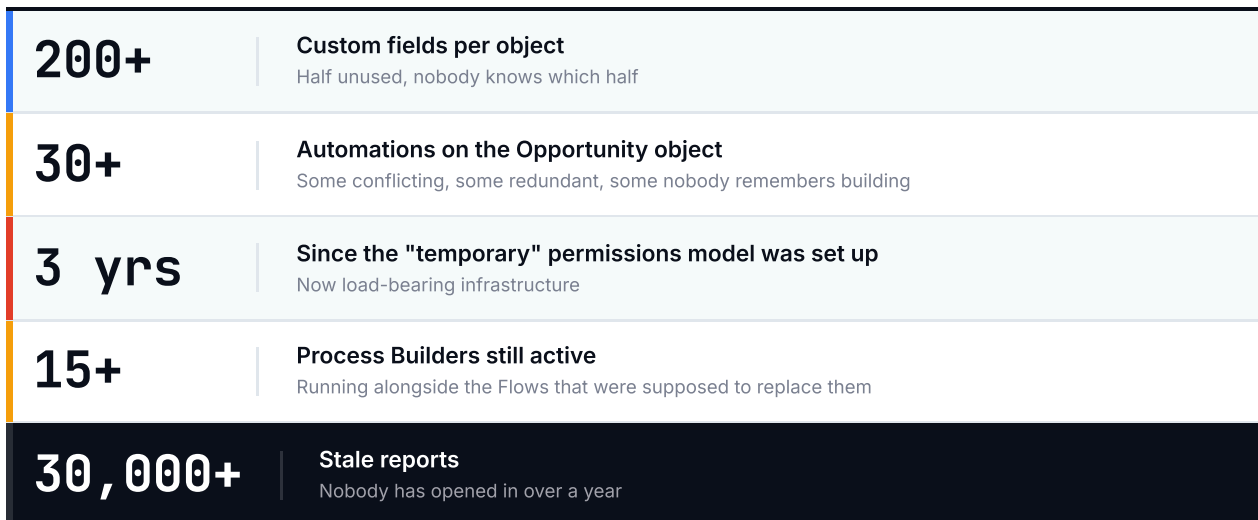


FIGURE 1 – THE AVERAGE SALESFORCE ORG TODAY

Technical debt isn't just annoying. It's costing real money. Slower deployments. Broken automations at 2 AM. Bad data poisoning pipeline forecasts. Admin burnout (and turnover). Projects that should take a week taking a month.

And the worst part? Leadership doesn't see it. They see a CRM that "works." They see an admin who seems to be busy but can't explain why a "simple" field addition took two sprints.

**This guide makes the invisible visible.** For admins: you'll get the exact frameworks, queries, AI prompts, and playbooks to audit, prioritize, and fix your org's technical debt. For leadership: you'll finally understand why "simple changes" aren't simple — and what it's costing you.



## Neil Sarkar

CTO & CO-FOUNDER, CLIENTELL

150+ Salesforce implementations. 6+ years in the ecosystem. Built AI agents that do Salesforce admin work from plain English. Before Clientell, scaled Salesforce practices at two VC-backed startups. Has personally audited hundreds of orgs and seen every flavor of technical debt imaginable.

[linkedin.com/in/neil-sarkar](https://www.linkedin.com/in/neil-sarkar) [x.com/NeilSarkr](https://x.com/NeilSarkr) [neil@getclientell.com](mailto:neil@getclientell.com)

# Table of Contents

CLIENTELL GUIDE  
2026 EDITION  
10 CHAPTERS

---

01	<b>What Is Salesforce Technical Debt</b> And Why Should Leadership Care	4
02	<b>The True Cost Calculator</b> Stop Guessing, Start Measuring	7
03	<b>The Technical Debt Audit</b> Where to Look: Queries, Tools & AI Prompts	10
04	<b>Automation Debt: The Silent Killer</b> Process Builders, Workflow Rules & Cascading Failures	13
05	<b>Field &amp; Metadata Debt</b> The 200-Field Problem	17
06	<b>Permission Debt</b> The Security Time Bomb	20
07	<b>Data Debt: The Foundation Problem</b> Duplicates, Inconsistency, Staleness	23
08	<b>The AI-Powered Cleanup Playbook</b> Tools and Approaches for Every Debt Type	26
09	<b>The 90-Day Debt Reduction Plan</b> Week-by-Week Checklist	29
10	<b>Making the Business Case</b> Speaking Leadership's Language	33

---

CHAPTERS

CLIENTELL  
**10**

AI PROMPTS

**15** +

SOQL QUERIES

**20** +

WORKSHEETS

**5**

# What Is Salesforce Technical Debt (and Why Should Leadership Care)

Every shortcut, workaround, and "temporary" fix that makes your org harder to change tomorrow than it is today.

---

THE ANALOGY

Credit card debt that compounds

TYPES

5 categories of technical debt

KEY INSIGHT

1 shortcut = \$15K+ over 4 years

# The Definition That Actually Makes Sense

Technical debt is every shortcut, workaround, and "temporary" fix that makes your Salesforce org harder to change tomorrow than it is today.

Think of it like credit card debt. Every quick fix — that Process Builder you built in 10 minutes instead of designing a proper Flow, that extra custom field you created because "we might need it," that profile with Modify All Data because debugging permissions was taking too long — is a charge on the credit card. And like real debt, it compounds.

THE COMPOUND INTEREST OF ONE SHORTCUT

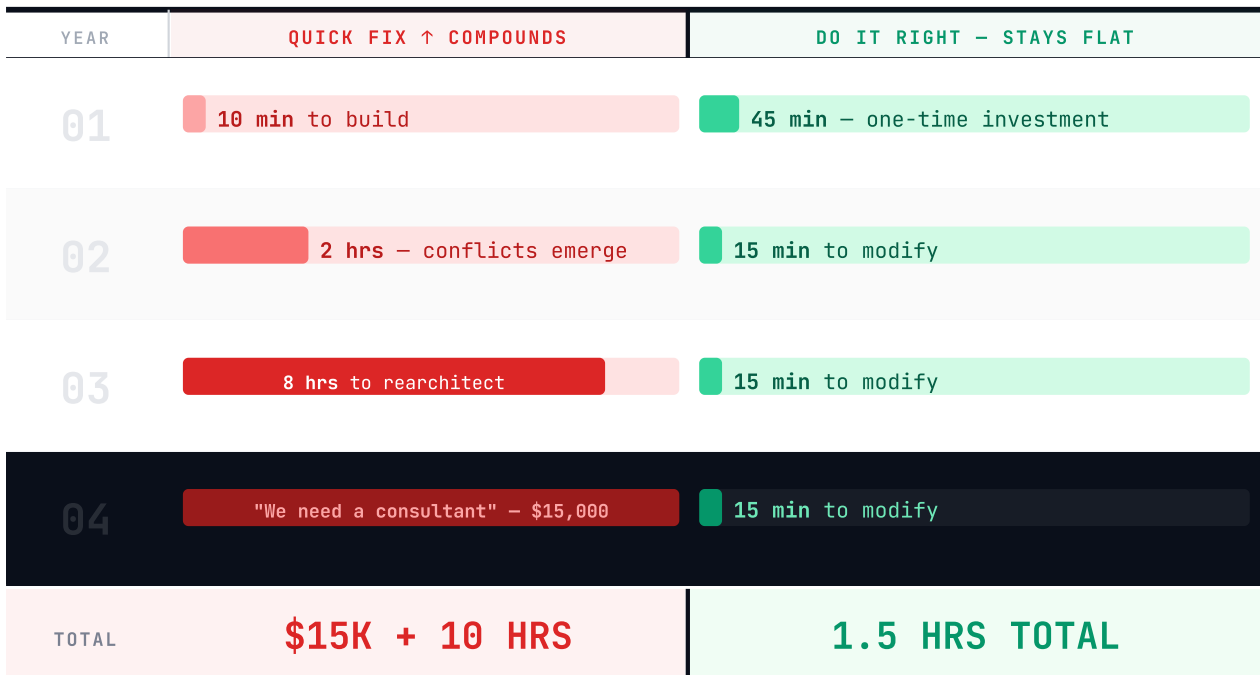


FIGURE 2 – THE COST OF ONE SHORTCUT OVER FOUR YEARS. MULTIPLY BY HUNDREDS OF DECISIONS.

One shortcut. One. Multiply that by the hundreds of decisions made over years, and you begin to see the real balance sheet.

# The Five Types of Salesforce Technical Debt

THE FIVE TYPES OF SALESFORCE TECHNICAL DEBT

DEBT TYPE · KEY SYMPTOM		RISK LEVEL
01	<b>Automation Debt</b> 3-4 tools fire on the same objects. Conflicts nobody can trace.	● HIGH RISK
02	<b>Field Debt</b> 200+ custom fields, 30-50% unused. No descriptions. No ownership.	● MED RISK
03	<b>Permission Debt</b> Only 34% migrated to permission sets. Modify All used as a shortcut.	● HIGH RISK
04	<b>Data Debt</b> Duplicates, stale records, inconsistent formats. Poisons pipeline data.	● MED RISK
05	<b>Documentation Debt</b> 78% of orgs have no formal docs. Knowledge leaves when the admin does.	● ONGOING

IMPACT: Slow deploys · Broken automations · Bad pipeline data · Admin burnout

FIGURE 3 – TAXONOMY OF SALESFORCE TECHNICAL DEBT

**FOR LEADERSHIP**

"We're paying for a \$200,000/year CRM that operates at 60% efficiency because of accumulated shortcuts. Every 'simple change' requires navigating around legacy configurations that nobody fully understands. This costs us [X hours/month] in admin time, [Y failed deployments/quarter], and delays every project by [Z weeks]. We can fix it, but it requires dedicated time and prioritization — not just more feature requests piled on top."

# The True Cost Calculator

Stop guessing. Start measuring. "Things are slow" doesn't get budget. "\$47,000 per year in wasted admin hours" does.

---

MID-MARKET COST

\$12K-\$25K/month

ENTERPRISE COST

\$50K+/month

CATEGORIES

5 cost categories

# The Technical Debt Cost Worksheet

## FIVE CATEGORIES OF TECHNICAL DEBT COST

COST CATEGORY · KEY METRIC		IMPACT
01	<b>Deployment Time Wasted</b> Admin spends 3–5× longer per deployment than in a clean org.	● 3-5× LONGER
02	<b>Automation Failures</b> Debt-heavy orgs see 4× more automation failures per month.	● 4× FAILURES
03	<b>Data Quality Impact</b> Bad CRM data costs companies 12% of annual revenue (Gartner).	● 12% REVENUE
04	<b>Admin Overtime &amp; Opportunity Cost</b> 60–70% of admin time goes to maintenance vs. new features.	● 60-70% MAINT.
05	<b>Risk &amp; Compliance Exposure</b> One breach or audit failure costs \$150K–\$1M+. Overpermissioned orgs are exposed.	● BREACH COST

TOTAL MONTHLY COST: \$12-25K for mid-market orgs · \$50K+ for enterprise (500+ users)

FIGURE 4 – THE FIVE COST CATEGORIES, RANKED BY TYPICAL IMPACT

### Category 1: Deployment Time Wasted

METRIC	YOUR NUMBER
Average deployments per month	---
Average time per deployment (hours)	---
Time that SHOULD take without conflicts/workarounds	---
Excess hours per deployment	---
Admin hourly rate (fully loaded)	\$---
<b>Monthly cost: (excess hrs x deployments x rate)</b>	<b>\$---</b>

Industry benchmark: Orgs with high technical debt spend 3-5x longer on deployments than clean orgs.

### Category 2: Automation Failures

METRIC	YOUR NUMBER
Automation failures per month	---
Average hours to diagnose and fix	---
Hours spent on "emergency" fixes outside business hours	---
Revenue-impacting failures	---
<b>Monthly cost: (total fix hours x rate) + revenue impact</b>	<b>\$---</b>

### Category 3: Data Quality Impact

METRIC	YOUR NUMBER
Estimated duplicate record rate	___%
Deals affected by bad data per quarter	---
Average deal size	\$___
Estimated revenue loss from bad data (conservative: 5% of pipeline)	\$___
<b>Monthly cost: (revenue loss / 3) + (cleanup hours x rate)</b>	<b>\$___</b>

Benchmark: Bad CRM data costs companies an average of 12% of their revenue (Gartner). Even a conservative 2-3% impact on a \$10M pipeline is \$200-300K/year.

### Category 4: Admin Overtime & Opportunity Cost

METRIC	YOUR NUMBER
Hours/month on "keeping the lights on" vs. new projects	---
Delayed projects due to tech debt	---
Estimated value of delayed projects per month	\$___
Admin overtime hours per month (evenings, weekends)	---
<b>Monthly cost: (overtime x rate) + delayed project value</b>	<b>\$___</b>

### Category 5: Risk & Compliance Exposure

METRIC	YOUR NUMBER
Users with Modify All Data who shouldn't have it	---
Months since last permission audit	---
Compliance frameworks (SOX, HIPAA, GDPR)	---
Estimated cost of a single data breach or audit failure	\$___
<b>Monthly risk exposure: (breach cost x probability)</b>	<b>\$___</b>

**\$12-25<sup>K</sup>**  
 Monthly cost for mid-market

**\$50<sup>K+</sup>**  
 Monthly cost for enterprise

**100<sup>+</sup>**  
 Org audits by Clientell

# The Technical Debt **Audit**: Where to Look

Don't try to fix everything at once. First, you need to know what you're dealing with. Think of it as a medical checkup — before treatment, you need a diagnosis.

# The Systematic Approach

## THE 6-STEP AUDIT PROCESS

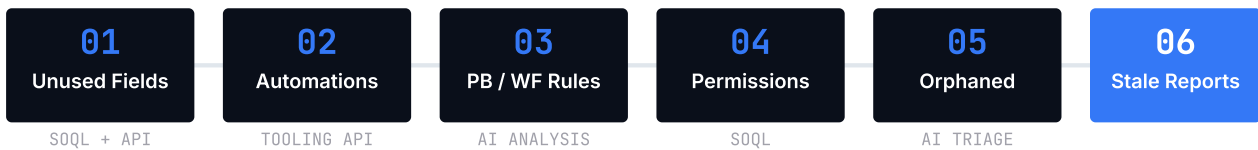


FIGURE 5 – THE SIX-STEP AUDIT PROCESS

### Step 1: Unused Fields

Fields that nobody uses are the most common form of tech debt. They clutter page layouts, confuse users, slow down data loads, and make it harder to understand your data model.

```
SELECT COUNT(Custom_Field_1__c), COUNT(Custom_Field_2__c), COUNT(Custom_Field_3__c)
FROM Opportunity
WHERE CreatedDate = LAST_N_DAYS:180
```

SOQL

#### AI PROMPT

I have [NUMBER] custom fields on [OBJECT]. Here are the field names and their population rates over the last 6 months:

[PASTE YOUR DATA]

Categorize each field as:

1. ACTIVE (>50% populated, used regularly)
2. LOW USAGE (10-50% populated, may be legacy)
3. LIKELY UNUSED (<10% populated, candidate for removal)
4. CRITICAL REGARDLESS (even low usage is expected – e.g., edge case fields)

For each LIKELY UNUSED field, suggest: safe to delete, archive first, or investigate further.

Include the dependency check I should run before deleting anything.

### Step 2: Conflicting Automations

This is where the real pain lives. Multiple automations firing on the same object, in unpredictable order, sometimes overwriting each other.

```
-- Flows
SELECT Definition.DeveloperName, ProcessType, Status, TriggerType
FROM FlowDefinitionView
WHERE ProcessType IN ('AutoLaunchedFlow', 'RecordTriggerFlow', 'Workflow')
AND Status = 'Active'

-- Workflow Rules
SELECT Name, TableEnumOrId
FROM WorkflowRule
WHERE TableEnumOrId = 'Opportunity'
```

TOOLING API

### Step 3: Redundant Process Builders and Workflow Rules

Salesforce officially retired Process Builder for new automations in 2023. If you still have active Process Builders, they're technical debt by definition.

AI PROMPT

I have [NUMBER] active Process Builders in my org. Here they are:

[LIST EACH WITH: Name, Object, What it does (brief)]

For each one, provide:

1. Can this be directly migrated to a Flow? (YES/NO/PARTIAL)
2. What's the migration complexity? (SIMPLE/MODERATE/COMPLEX)
3. Are there any gotchas in the migration?
4. Priority to migrate (based on object importance and conflict risk)

Create a migration plan ordered by: quick wins first, then critical-path items, then nice-to-haves.

### Step 4: Over-Permissioned Profiles

SOQL

```
-- Users with Modify All Data
SELECT Id, Name, Profile.Name, LastLoginDate
FROM User
WHERE IsActive = true
AND Profile.PermissionsModifyAllData = true

-- Permission sets granting dangerous permissions
SELECT Label, PermissionsModifyAllData, PermissionsViewAllData,
       PermissionsManageUsers, PermissionsAuthorApex
FROM PermissionSet
WHERE PermissionsModifyAllData = true
OR PermissionsViewAllData = true
```

### Step 5: Orphaned Metadata

Components that exist but aren't connected to anything. Fields not on any page layout. Flows that are inactive but not deleted. Apex classes with 0% test coverage and no references.

AI PROMPT

Help me identify orphaned metadata in my Salesforce org. Here's what I've extracted:

Custom fields not on any page layout: [LIST]  
 Inactive Flows: [LIST WITH LAST MODIFIED DATE]  
 Apex classes with 0% coverage: [LIST]  
 Reports not viewed in 12+ months: [COUNT]

For each category:

1. What's safe to delete immediately?
2. What needs dependency checking first?
3. What should be archived (not deleted)?
4. What's the process for safe removal?

Give me a prioritized cleanup list starting with the lowest-risk, highest-impact items.

### Step 6: Stale Reports

# Automation Debt: The **Silent Killer**

If fields are the atoms of your Salesforce org, automations are the nervous system. And in most orgs, that nervous system has been rewired so many times that nobody fully understands how a signal gets from A to B.

---

ROOT CAUSE

3-4 automation tools on same objects

#1 SOURCE

Process Builders (retired 2023)

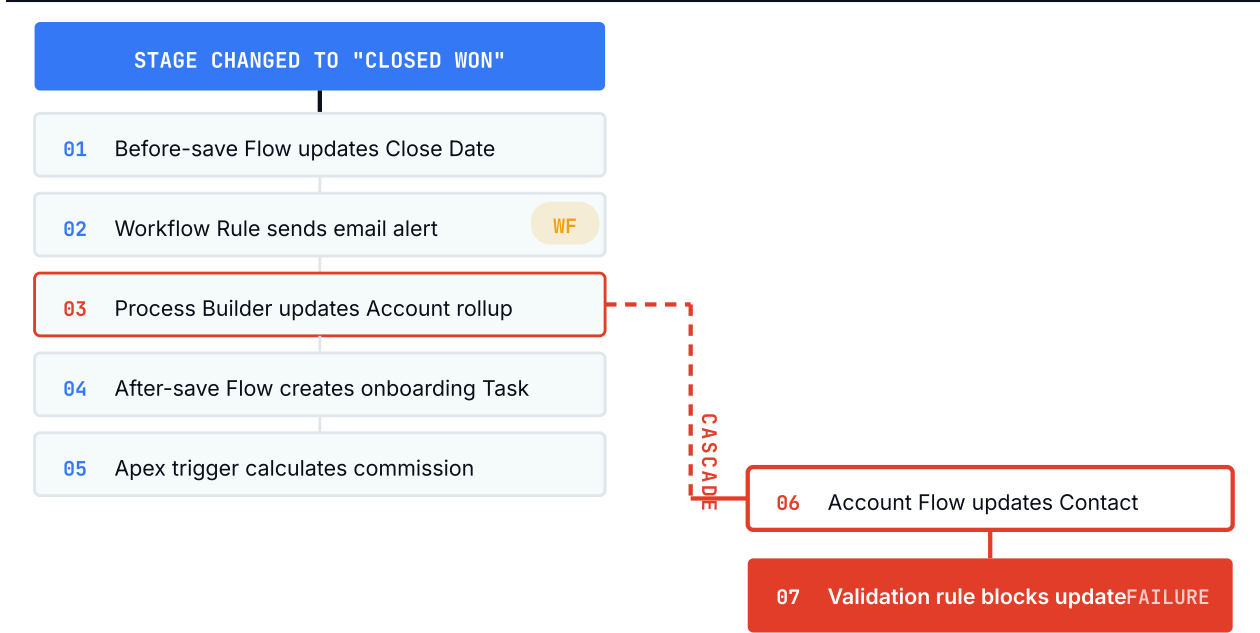
IMPACT

#1 source of admin burnout

# The Problem Nobody Talks About

Here's a real scenario we see in almost every audit. A sales rep changes the Stage on an Opportunity to "Closed Won." Here's what fires:

WHAT ACTUALLY HAPPENS WHEN A REP CHANGES STAGE TO "CLOSED WON"



"An unhandled fault has occurred." Rep pings Slack. You spend 90 min debugging.

FIGURE 6 – A REAL AUTOMATION CASCADE FAILURE FROM A SINGLE STAGE CHANGE

**This is automation debt.** And it's the single biggest source of admin burnout.

*The average Enterprise org has 3-4 automation tools running simultaneously on key objects. Each one was built with good intentions. Together, they create an execution order nobody fully controls.*

-- Clientell Org Audit Data, 2026

# The Automation Audit

For your top 5 objects (usually Opportunity, Account, Contact, Lead, Case), build this table:

TYPE	NAME	TRIGGER	WHAT IT DOES	CREATED
Record-Triggered Flow	Opp_Stage_Update	After Save	Creates Task, sends email	2024-01
Process Builder	Update_Account_Revenue	On Create/Edit	Updates Account.Total_Rev	2021-06
Workflow Rule	Closed_Won_Alert	On Create/Edit	Email alert	2020-03
Apex Trigger	OpportunityTrigger	After Insert/Update	Commission calc	2022-09

AI PROMPT

Here is every automation on our Opportunity object:

[PASTE YOUR TABLE]

Analyze this for:

- ORDER OF EXECUTION CONFLICTS: Which automations could fire in a problematic sequence?
- FIELD COLLISIONS: Which automations update the same field(s)?
- RECURSION RISK: Which could trigger each other in a loop?
- REDUNDANCY: Which do overlapping or identical things?
- LEGACY DEBT: Which use deprecated tools (Process Builder, Workflow Rules)?

For each issue found, rate it RED (will cause failures), YELLOW (may cause unexpected behavior), or GREEN (low risk but should be cleaned up).

Then recommend a target state: what should the automation landscape look like after cleanup?

# The Process Builder Problem

Process Builders are the #1 source of automation debt in most orgs. They're being retired (Salesforce stopped supporting new ones in 2023), they run in the same transaction as Flows but in a different order, they can't be version-controlled, and they're invisible to many debugging tools.

## PROCESS BUILDER MIGRATION DECISION TREE



# Workflow Rules: The Cockroaches of Salesforce

Workflow Rules are even more legacy than Process Builders. They've been deprecated since 2022 but they're still running in nearly every org we audit.

Common Workflow Rule debt: email alerts that should be Flows, field updates that conflict with Flow field updates, outbound messages to integrations that were replaced years ago, and time-dependent actions that nobody remembers exist.

```
SELECT Name, TableEnumOrId
FROM WorkflowRule
WHERE ManageableState = 'unmanaged'
```

SOQL

AI PROMPT

Here are all Workflow Rules in my org: [LIST]

For each, classify as:

- MIGRATE (has active business value, needs to become a Flow)
- DEACTIVATE (no longer needed, safe to turn off after verification)
- INVESTIGATE (unclear purpose, needs someone to confirm before touching)

Provide a safe deactivation checklist for each DEACTIVATE candidate.

AI PROMPT

I need to migrate this Process Builder to a Flow. Here are the details:

Name: [NAME]

Object: [OBJECT]

Criteria Groups: [DESCRIBE EACH]

Actions per criteria group: [DESCRIBE EACH]

Does it interact with other automations on this object? [YES/NO – DESCRIBE]

Does it call Apex Invocable Actions? [YES/NO]

Provide:

1. A Flow design (trigger type, entry criteria, elements needed)
2. Any gotchas in the migration (things that work differently in Flows vs. PB)
3. Testing plan (what to verify before deactivating the Process Builder)
4. Rollback plan (how to revert if something breaks)
5. Step-by-step migration instructions

## FOR LEADERSHIP

Your team is running the equivalent of Windows XP alongside Windows 11. Process Builders and Workflow Rules are legacy technology that Salesforce is actively sunsetting. Every month you wait to migrate adds risk and cost. The migration itself isn't glamorous, but it's one of the highest-ROI projects your admin can do.

# Field & Metadata Debt: The 200-Field Problem

Every field starts with good intentions. Nobody ever deletes them.

---

AVERAGE FIELDS

200+ custom per object

UNUSED

30-50% of all custom fields

SAFE DELETE RULE

Remove from layouts, wait 2-4 weeks

# Death by a Thousand Fields

A VP says "I need a field for X." Marketing says "We need to track Y." A consultant creates 30 fields for a project that gets shelved after two months. The result: objects with 200, 300, even 500+ custom fields.

SYMPTOM	IMPACT
200+ custom fields on Opportunity	User confusion, slow page loads, cluttered layouts
30-50% of fields with <5% population	Wasted screen real estate, misleading reports
Duplicate fields (same data, different names)	Inconsistent reporting, conflicting data
Fields with no description	New admins can't understand the data model
Picklist values not updated in 2+ years	Bad data, workarounds via "Other" field
Fields created by consultants for ended projects	Nobody knows if they're safe to delete

## The Field Audit Process

### Step 1: Get the Full Field Inventory

```
SELECT DeveloperName, Description, NamespacePrefix, CreatedDate,  
       LastModifiedDate, Metadata  
FROM CustomField  
WHERE TableEnumOrId = 'Opportunity'  
AND NamespacePrefix = null  
ORDER BY CreatedDate ASC
```

TOOLING API

### Step 2: Measure Field Population

```
SELECT  
  COUNT(Id) Total_Records,  
  COUNT(Field_1__c) Field_1_Populated,  
  COUNT(Field_2__c) Field_2_Populated,  
  COUNT(Field_3__c) Field_3_Populated  
FROM Opportunity  
WHERE CreatedDate = LAST_N_DAYS:365
```

SOQL

Pro tip: SOQL COUNT(field) counts non-null values. You can check 25 fields per query. For an object with 200 fields, that's 8 queries.

## AI PROMPT

Here is a field inventory for our [OBJECT] object. For each field I'm including: name, description (if any), population rate, last modified date, and known dependencies.

[PASTE DATA]

Categorize each field into:

1. KEEP – Actively used, well-populated, has dependencies
2. CONSOLIDATE – Overlaps with another field, merge data and delete one
3. ARCHIVE – Low usage but might have historical value, hide from layouts but don't delete yet
4. DELETE – Zero usage, no dependencies, safe to remove
5. INVESTIGATE – Unclear purpose, needs business owner confirmation

For CONSOLIDATE candidates, identify which fields should be merged and suggest a data migration approach.

For DELETE candidates, provide a safe deletion checklist (dependency check, layout removal, Flow check, report check).

## The Metadata Sprawl Problem

Fields aren't the only metadata that accumulates. Consider: Record Types nobody uses (but can't delete because they're referenced in profiles). Page Layouts that are copies of copies ("Layout v2", "Layout FINAL", "Layout NEW"). Custom Objects created for proof-of-concepts that never went anywhere. Custom Tabs nobody has clicked in years.

## AI PROMPT

Here is a summary of our Salesforce metadata inventory:

Custom Objects: [NUMBER] (list objects with <100 records)  
 Record Types per object: [LIST OBJECTS WITH 4+ RECORD TYPES]  
 Page Layouts: [NUMBER] (list any with "v2", "FINAL", "NEW", "OLD", "COPY" in the name)  
 Custom Tabs: [NUMBER]  
 Custom Apps: [NUMBER]

For each category, help me identify:

1. What's clearly safe to clean up?
2. What needs dependency checking first?
3. What's the recommended approach (delete, merge, archive)?
4. What's the order of operations (what to clean first to unblock cleaning other things)?

## PRO TIP

Never delete a field the same day you decide it's unused. Move it off all page layouts first. Wait 2-4 weeks. If nobody notices, it's safe. If someone panics, you just add it back to the layout — no data lost.

*You're halfway through. If you're reading this and mentally counting the custom fields on your Opportunity object right now, you're exactly who this guide is for.*

# Permission Debt: The Security Time Bomb

The most dangerous type of technical debt because it's not just inefficient — it's a security risk.

---

OVER-PERMISSIONED

72% of orgs have non-admin MAD

MIGRATION

Only 34% started profile migration

AUDIT CADENCE

89% have no regular reviews

# The Debt Nobody Audits Until It's Too Late

<h2>72%</h2> <p>Orgs with non-admin Modify All Data</p>	<h2>23</h2> <p>Average permission sets with overlap</p>	<h2>34%</h2> <p>Started profile-to- permset migration</p>	<h2>89%</h2> <p>No regular permission review cadence</p>
-------------------------------------------------------------	-------------------------------------------------------------	---------------------------------------------------------------	--------------------------------------------------------------

## Step 1: Find Your Highest-Risk Users

```

-- Users with Modify All Data (should be admins ONLY)
SELECT Id, Name, Profile.Name, UserRole.Name, LastLoginDate
FROM User
WHERE IsActive = true
AND Profile.PermissionsModifyAllData = true
ORDER BY Profile.Name

-- Users with View All Data
SELECT Id, Name, Profile.Name
FROM User
WHERE IsActive = true
AND Profile.PermissionsViewAllData = true

```

SQL

## Step 2: Identify Over-Permissioned Profiles

```

SELECT Parent.Profile.Name, SubjectType,
       PermissionsModifyAllRecords, PermissionsViewAllRecords
FROM ObjectPermissions
WHERE PermissionsModifyAllRecords = true
AND Parent.IsOwnedByProfile = true

```

SQL

## Step 3: Find Permission Set Overlap

```

SELECT PermissionSet.Label, Assignee.Name, Assignee.Profile.Name
FROM PermissionSetAssignment
WHERE PermissionSet.IsOwnedByProfile = false
ORDER BY Assignee.Name

```

SQL

Here is our current permission structure:

Profiles in use: [LIST WITH USER COUNT PER PROFILE]  
 Permission Sets: [LIST WITH DESCRIPTION AND USER COUNT]  
 Permission Set Groups: [LIST IF ANY]

Please analyze and provide:

1. Which profiles have excessive permissions?
2. Which permission sets overlap significantly?
3. Which users have permission combinations that create security risks?
4. A recommended target state: minimum profiles + permission set groups per job function
5. A migration sequence that won't break access for active users

# The Profile-to-Permission-Set Migration

This isn't optional anymore. Salesforce has announced the deprecation of profile-based permissions. But only 34% of orgs have started the migration.

## THE FOUR-PHASE PERMISSION MIGRATION

PHASE · ACTIVITIES		TIMELINE
01	<b>Discovery</b> Export profiles, map used permissions. Identify baseline access patterns.	1 WEEK
02	<b>Design</b> Build permission set groups and access matrix. One permission set per role.	1-2 WEEKS
03	<b>Pilot</b> Migrate simplest profile first. Side-by-side access testing in sandbox.	2 WEEKS
04	<b>Rollout</b> One profile at a time. 48-hour validation window per batch before proceeding.	4-8 WEEKS

TOTAL TIMELINE: 8-13 WEEKS

Never migrate all profiles at once

FIGURE 8 – PROFILE-TO-PERMISSION-SET MIGRATION PHASES

AI PROMPT

I need to migrate from profile-based permissions to permission sets. Here are my current profiles:

[LIST EACH PROFILE WITH: Name, User Count, Key Permissions, Objects Accessed]

Our compliance requirements: [SOX/HIPAA/GDPR/NONE]  
 Salesforce Edition: [EDITION]  
 Do we use Permission Set Groups yet? [YES/NO]

Create a phased migration plan including:

1. Which profile to migrate first (lowest risk, best pilot)
2. Permission Set Group design for each job function
3. The minimum-access profile template
4. Validation queries to confirm no access was lost
5. Rollback procedure for each step
6. Timeline estimate

**FOR LEADERSHIP**

Permission debt is a compliance risk. If you're subject to SOX, HIPAA, or GDPR, an auditor asking "who has access to what?" and getting a shrug is a finding. The migration to permission sets is inevitable — Salesforce is forcing it. The question is whether you do it on your timeline or theirs.

# Data Debt: The Foundation Problem

Your CRM is only as good as its data. You can have perfect automations, clean metadata, and tight permissions. But if the data is a mess, none of it matters.

---

DUPLICATE RATE

10-25% average across Accounts

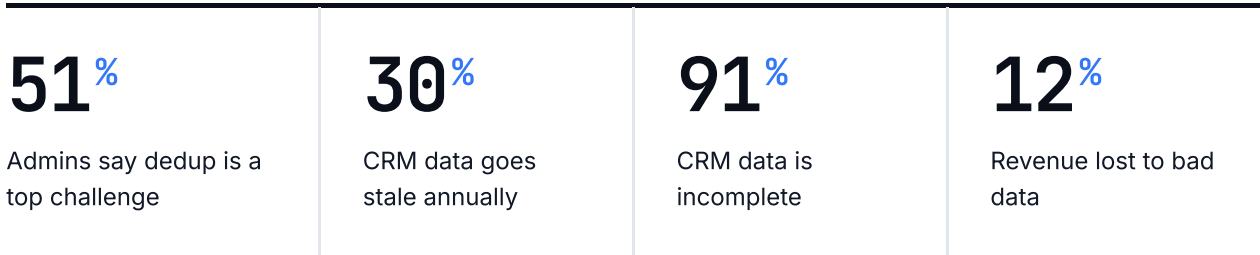
DATA STALENESS

30% goes stale every year

REVENUE IMPACT

12% lost to bad data (Gartner)

# The Four Types of Data Debt



FOUR TYPES OF DATA DEBT

TYPE · SYMPTOM	BUSINESS IMPACT
<p>01 <b>Duplicate Records</b>                      "Acme Inc" = "ACME" = "Acme, Inc." — each appears as a unique record.</p>	<p>● 20-30% INFLATED</p>
<p>02 <b>Inconsistent Formats</b>                      "US" vs "USA" vs "United States." Segment filters break. Reports disagree.</p>	<p>● BREAKS FILTERS</p>
<p>03 <b>Incomplete Records</b>                      Leads without email. Accounts without industry. Scoring models fail silently.</p>	<p>● MISSED INSIGHTS</p>
<p>04 <b>Stale Data</b>                      Contacts who left, deals stuck 6+ months. Forecasts built on fiction.</p>	<p>● ACTIVELY MISLEADS</p>

ROOT CAUSE: bad input habits · no dedup rules · no field ownership · no purge policy

FIGURE 9 – THE FOUR TYPES OF DATA DEBT AND THEIR BUSINESS IMPACT

# The Data Quality Audit

## Step 1: Measure Your Duplicate Rate

AI PROMPT

I need to estimate my Salesforce duplicate rate. Here are my numbers:

Total Accounts: [NUMBER] | Total Contacts: [NUMBER] | Total Leads: [NUMBER]

I've exported a sample of 500 Accounts. Here are potential duplicate clusters:  
 [PASTE SAMPLE – even 20-30 examples is enough]

Based on this sample:

1. What's my estimated duplicate rate?
2. What matching criteria should I use for dedup?
3. What's the recommended approach: native Matching Rules, third-party, or manual?
4. What's the merge strategy (which record is the "master")?
5. What data do I need to preserve from the duplicate before merging?

### Step 3: Find Stale Records

```

-- Opportunities stuck in mid-funnel
SELECT Id, Name, StageName, LastModifiedDate, Owner.Name
FROM Opportunity
WHERE IsClosed = false
AND LastModifiedDate < LAST_N_DAYS:90
ORDER BY LastModifiedDate ASC

-- Contacts with bounced emails
SELECT Id, Name, Email, Account.Name
FROM Contact
WHERE HasOptedOutOfEmail = true OR Email = null

```

SQL

## Data Cleanup With AI

AI PROMPT

I need to standardize data in my Salesforce org. Here are the fields with issues:

1. Country field: contains "US", "USA", "United States", "U.S.", "America", etc.
2. State field: mix of abbreviations and full names
3. Phone field: formats include (555) 555-1234, 555-555-1234, 5555551234, +1 555 555 1234
4. Industry picklist: users type in "Other" and put the real answer in a notes field

For each field:

1. Define the standard format we should use
2. Write the Salesforce formula or Flow logic to standardize incoming data
3. Provide a Data Loader approach for cleaning existing records
4. Suggest validation rules to prevent future bad data
5. Estimate the effort (hours) for each cleanup task

AI PROMPT

Help me create a data quality scoring system for our Salesforce org.

Key objects: Account, Contact, Lead, Opportunity

Critical fields per object:

- Account: [LIST YOUR MUST-HAVE FIELDS]
- Contact: [LIST]
- Lead: [LIST]
- Opportunity: [LIST]

Design:

1. A completeness score (% of critical fields populated) per record
2. A formula field or Flow that calculates and stores this score
3. Thresholds (what score = "good", "needs attention", "critical")
4. A dashboard showing data quality trends over time
5. Alerts when quality drops below threshold

FOR LEADERSHIP

Bad data isn't an admin problem — it's a revenue problem. When your pipeline report says \$10M but 20% of those deals are duplicates or stale, you're making decisions based on a number that's off by \$2M. The cost of data cleanup is a fraction of the cost of bad forecasting.

# The AI-Powered Cleanup Playbook

You've identified the debt. Now let's fix it. This chapter is your toolbox — specific tools, AI prompts, and step-by-step approaches for each type of technical debt.

---

GENERAL AI

ChatGPT, Claude, Gemini

SF-SPECIFIC

Hubbl, Sweep, Elements,  
OrgCheck

CLIENTELL

Full org scan in 2 minutes

# The AI Tools Landscape

AI TOOLS FOR SALESFORCE DEBT CLEANUP

TOOL	BEST USE CASE	COST	DEPTH
ChatGPT / Claude / Gemini	Analysis, planning, writing SOQL	\$0-20 / mo	SHALLOW
Hubbl Diagnostics	Org benchmarking, health scores	\$79-849 / mo	MODERATE
Sweep	Visual canvas, change management	\$1-2K / mo	MODERATE
OrgCheck / HappySoup	Quick health check, dependency map	Free	SHALLOW
<b>Clientell AI Agents</b>	Full scan, dependency map, remediation plan	Custom	<b>FULL SCAN</b>

Depth = coverage of automation + fields + permissions + data + docs

FIGURE 10 – AI TOOLS FOR SALESFORCE TECHNICAL DEBT CLEANUP

## The "Do It With AI" Approach

**01**

### Export

Your data using the SOQL queries from Chapters 3-7

**02**

### Paste

Into ChatGPT or Claude with the specific prompts from each chapter

**03**

### Get

A categorized, prioritized action plan from the AI

**04**

### Validate

The AI's recommendations against your org knowledge

**05**

### Execute

In sandbox first, always. Test with real users before production.

**06**

### Monitor

For 48-72 hours after deployment. Have a rollback plan ready.

# Playbook by Debt Type

## Automation Debt Cleanup

#	ACTION	TOOL	TIME
1	Inventory all automations per object	Tooling API + AI	2-4 hrs
2	Map conflicts and redundancies	AI analysis	1-2 hrs
3	Migrate Process Builders (simple ones)	SF Migration Tool	2-4 hrs each
4	Redesign complex automations	AI + Flow Builder	4-8 hrs each
5	Test everything in sandbox	Manual	4-8 hrs

## Field Debt Cleanup

#	ACTION	TOOL	TIME
1	Export field inventory + measure population	Tooling API + SOQL	3-4 hrs
2	Check dependencies for low-usage fields	Tooling API / Clientell	2-4 hrs
3	Categorize (keep/consolidate/archive/delete)	AI analysis	2 hrs
4	Remove from page layouts, wait 2-4 weeks	Manual	2-4 hrs
5	Delete confirmed-unused fields	Manual	1-2 hrs

## Permission Debt Cleanup

#	ACTION	TOOL	TIME
1	Audit current permissions	SOQL + AI	4-8 hrs
2	Design target permission model	AI + human design	8-16 hrs
3	Pilot with one profile	Manual + validation	1-2 weeks
4	Roll out remaining profiles	Manual + validation	4-8 weeks

## Data Debt Cleanup

#	ACTION	TOOL	TIME
1	Measure duplicate rate + completeness	SOQL + AI	3-6 hrs

# The 90-Day Debt Reduction Plan

Three phases. Twelve weeks. From diagnosis to structural improvement.

---

PHASE 1

Audit & Prioritize (Month 1)

PHASE 2

Quick Wins (Month 2)

PHASE 3

Structural Fixes (Month 3)

# Phase 1: Audit and Prioritize (Month 1)

The goal of Month 1 isn't to fix anything. It's to understand exactly what you're dealing with and build the case for fixing it.

## 90-DAY TIMELINE OVERVIEW

	PHASE • FOCUS AREAS	WEEKS
01	<b>Month 1 • Audit &amp; Prioritize</b> Health check → Full audit → Prioritize backlog → Get leadership buy-in.	1-4
02	<b>Month 2 • Quick Wins</b> Automation cleanup → Unused fields → Permission fixes → Data dedup.	5-8
03	<b>Month 3 • Structural Fixes</b> Architecture redesign → Permission migration → Data quality overhaul.	9-12

Weeks 1-4 audit • Weeks 5-8 execution • Weeks 9-12 structural • Total: 90 days

### WEEK 1 The Quick Health Check

- Run Salesforce Optimizer (Setup > Optimizer)
- Count automations per key object (Opportunity, Account, Contact, Lead, Case)
- Count custom fields per key object
- Run the "Modify All Data" query from Chapter 6
- Check how many active Process Builders and Workflow Rules you have

Estimated time: 4-6 hours

### WEEK 2 The Deep Audit

- Complete field population analysis (Chapter 5 queries)
- Map all automations per object (Chapter 4 table)
- Run permission audit queries (Chapter 6)
- Measure duplicate rate (Chapter 7)
- Measure field completeness rates (Chapter 7)

Estimated time: 12-16 hours

### WEEK 3 Prioritization

CLIENTELL

- Fill out the Cost Calculator worksheet (Chapter 2)

## Phase 2: Quick Wins and Critical Fixes (Month 2)

Now you fix things. Start with the items that are highest impact and lowest effort.

### WEEK 5 Automation Quick Wins

---

- Deactivate Workflow Rules that are clearly redundant (after sandbox testing)
- Deactivate Process Builders that have Flow equivalents already running
- Fix any automation conflicts causing active errors
- Document what you deactivated and why

### WEEK 6 Field and Metadata Quick Wins

---

- Remove unused fields from all page layouts (don't delete yet — just hide)
- Delete duplicate/test page layouts (the "v2" and "FINAL FINAL" ones)
- Clean up report folders (archive reports not viewed in 12+ months)
- Update field descriptions for your most-used custom fields

### WEEK 7 Permission Quick Wins

---

- Remove Modify All Data from any non-admin profile
- Create a "Bypass\_Validation\_Rules" custom permission (for Flows)
- Remove managed package licenses from inactive users
- Audit and consolidate overlapping permission sets

### WEEK 8 Data Quick Wins

---

- Set up Matching Rules and Duplicate Rules (if not already active)
- Merge the worst duplicate clusters (top 50-100 accounts)
- Create validation rules for critical formatting (phone, email, state)
- Archive stale Opportunities (closed-lost anything stuck > 6 months)

**Month 2 Deliverable:** 15-25 quick wins completed, measurable reduction in automation failures and deployment time, updated leadership on progress.

# After 90 Days: The Maintenance Cadence

The worst thing you can do is clean everything up and let it drift back. Establish these recurring practices:

## RECURRING MAINTENANCE CADENCE

	CADENCE · TASK SUMMARY	EFFORT
01	<b>Weekly · 15 min</b> Review automation error logs. Check deployment queue for stuck items.	● LOW EFFORT
02	<b>Monthly · 2 hrs</b> Field queries, automation review, merge duplicates, data quality check.	● CORE CADENCE
03	<b>Quarterly · ½ day</b> Permission + license audit. Automation inventory. Report to leadership.	● STRATEGIC
04	<b>Per ReLease · 4 hrs</b> SF release notes, deprecated features, sandbox testing, update docs.	● 3× / YEAR

SCHEDULE NOTE: Put these on the calendar — not a to-do list. Calendar blocks don't get deprioritized

FIGURE 11 — THE MAINTENANCE CADENCE THAT PREVENTS DEBT FROM RETURNING

### PRO TIP

Put these on the calendar. Not a to-do list — the calendar. To-do lists get deprioritized when a VP needs a report. Calendar blocks don't.

# Making the Business Case to Leadership

The hardest part of fixing technical debt isn't the technical work. It's getting permission to do it.

# Speaking Their Language

Leadership hears "technical debt" and thinks "admin wants to play with the system instead of building what we asked for." You need to translate the problem into language they understand: money, risk, speed, and competitive advantage.

## The Three Narratives That Work

THREE NARRATIVES THAT GET LEADERSHIP BUY-IN

	NARRATIVE · KEY PITCH	AUDIENCE
01	<b>The Money Story</b> "\$[X]/month on workarounds. A 90-day cleanup cuts that 40–60%. That's [X]× return."	● CFO / COO
02	<b>The Risk Story</b> "[N] users have Modify All Data. Deprecated tools could break any release. Auditors find gaps."	● CISO / LEGAL
03	<b>The Speed Story</b> "Every project is 30–50% slower. We deliver 60% of roadmap because 40% is maintenance."	● VP / CRO

KEY: Match narrative to audience – never pitch ROI to CISO or risk to CFO.

FIGURE 12 – MATCH YOUR NARRATIVE TO YOUR AUDIENCE

## The One-Page Executive Brief Template

SUBJECT: SALESFORCE TECHNICAL DEBT – IMPACT ASSESSMENT AND 90-DAY FIX PLAN

### The Problem (2 sentences)

Our Salesforce org has accumulated [X years] of technical shortcuts that are now costing us \$[AMOUNT]/month in wasted time, failed automations, and delayed projects. This is fixable, but it requires [X hours/month] of dedicated admin time for 90 days.

### The Cost (bullet points)

- \$[X]/month in deployment delays
- \$[X]/month in automation failures
- \$[X]/month in data quality impact
- [X] users with excessive data access (compliance risk)

### The Fix

Month 1: Audit and prioritize. Month 2: Quick wins (15–20 highest-impact items). Month 3: Structural improvements (migrate automations, redesign permissions).

### The Ask

[X] hours per week dedicated to debt reduction (not "between other tasks"). Permission to prioritize cleanup over [specific lower-priority project].

CLIENTELL

### The Return

40–60% reduction in monthly tech debt cost within 90 days. Faster delivery on every future project.

# Handling Common Objections

OBJECTION	RESPONSE
"We don't have time for this."	"We don't have time NOT to do this. Every week we wait, we spend [X] hours on workarounds. The cleanup pays for itself in reduced maintenance within [X] weeks."
"Can't we just hire a consultant?"	"We can. A consultant engagement typically costs \$15-30K. Alternatively, we can do 80% in-house with [X] hours of dedicated time, using AI tools to accelerate. The 20% that needs expert help is where a consultant adds the most value."
"Is this really costing us that much?"	"Here's the worksheet. [Present the Cost Calculator.] These are conservative estimates. The real number is likely higher because we're not counting opportunity cost."
"Can't we just rebuild from scratch?"	"A clean re-implementation costs \$150-300K and takes 6-12 months. The 90-day cleanup approach fixes 80% of the issues for 5% of that cost."
"What if we break something?"	"Every change is tested in sandbox first. We have rollback plans for each step. The bigger risk is NOT fixing it — we're one Salesforce release away from a deprecated Process Builder breaking in production."

*You're not asking for a favor. You're proposing a project with clear ROI. Frame it that way. Bring numbers, not complaints. The admin who quantifies the problem and proposes the solution is the one who gets promoted to Salesforce Architect.*

-- Advice for every Salesforce admin reading this guide

WHAT COMES NEXT

# This Guide Shows You What to Fix. Our AI Shows You Everything.

You now have the frameworks, queries, AI prompts, and playbooks to audit and fix your org's technical debt. That's real, usable knowledge.

But finding every issue manually takes 40-80 hours. Our AI scans your entire org in 2 minutes. Every field, every automation, every permission, every dependency.

[Book Your Free 15-Minute Org Audit](#)

NO SALES PITCH. NO FOLLOW-UP SPAM. JUST ANSWERS.

**2<sup>min</sup>**

Full org scan

**1,800<sup>+</sup>**

Metadata files analyzed

**\$<sup>0</sup>**

You keep the results

# Clientell

Salesforce Partner · Salesforce Startup Program

Salesforce & AI consultants who help companies fix broken orgs, build intelligent automation, and actually use the platform they're paying for.

<h2>2,000+</h2> <p>Admins use Clientell's AI for real Salesforce work</p>	<h2>150+</h2> <p>Salesforce implementations completed</p>	<h2>100+</h2> <p>Org audits performed</p>
---------------------------------------------------------------------------	-----------------------------------------------------------	-------------------------------------------

## ABOUT THE AUTHOR



### Neil Sarkar

CTO & CO-FOUNDER, CLIENTELL

150+ Salesforce implementations. 6+ years in the ecosystem. Built AI agents that do Salesforce admin work from plain English. Before Clientell, scaled Salesforce practices at two VC-backed startups. Has personally audited hundreds of orgs and seen every flavor of technical debt imaginable.

[linkedin.com/in/neil-sarkar](https://linkedin.com/in/neil-sarkar) · [x.com/NeilSarkar](https://x.com/NeilSarkar) · [substack.com/@thisisneilsarkar](https://substack.com/@thisisneilsarkar) · [neil@getclientell.com](mailto:neil@getclientell.com)

## METHODOLOGY

Statistics cited in this guide are drawn from Salesforce ecosystem surveys (Mason Frank, Salesforce Ben), Gartner research on CRM data quality, and Clientell's own audit data from 100+ org assessments conducted between 2024-2026. Cost estimates reflect median values from mid-market and enterprise organizations (200-500+ Salesforce users). Individual org costs may vary based on complexity, user count, and industry.

