THE CUSTOMER FEEDBACK LOOP · SUPPORT

How to build a help center that fixes itself.

A help center rewritten once a year is out of date for most of that year. The fix is not a bigger rewrite. It is a loop.

01
Detect
Find where the docs fail.

02
Diagnose
Name the gap.

03
Fix
One move per gap.

04
Verify
Measure the topic.

00 · THE PROBLEM

Why help centers stop working.

A help center rewritten once a year is out of date for most of that year. Most self-service attempts fail, usually because customers cannot find the right content. And AI agents now answer customers from your docs, so every gap in them becomes a gap in every assistant that reads them.

14%	43%	61%	51%
of service issues fully resolved in self-service	of failures: customers could not find relevant content	of service leaders have a backlog of articles to edit	of service journeys now start on third-party platforms

Sources: Gartner, Aug 2024 · CX Dive · Gartner, Dec 2024 · Gartner, Jul 2025

THE MODEL

The fix is a loop, not a bigger rewrite.

01 · p.03

Detect

Find where the docs fail from signals customers already leave.

02 · p.04

Diagnose

Group by question, size by customers, name the gap.

03 · p.05

Fix

One move per type of gap, one owner each.

04 · p.06

Verify

Measure the topic. Not fixed? Back to Detect.

05 · MEMORY A record of every gap, fix and result, so the loop never proposes the same fix twice and can tell a new problem from an old one that came back.

STEP 01 · DETECT

Detect where the docs fail.

Customers already mark the gaps for you. Each signal below is someone who tried self-service, did not get an answer, and showed it.

SIGNAL	WHAT IT TELLS YOU	WHERE TO GET IT
Failed searches	The words customers use, and whether any article matches them	Help center search logs (Zendesk Search dashboard)
Article downvotes	An article exists, but it did not answer the question	Article feedback, with the comment
Read, then ticketed	The strongest signal: they found the right page and it still failed	Help center sessions joined to ticket requesters
Bot escalations	Where your AI agent runs out of knowledge	AI agent transcripts (Intercom flags unresolved Fin chats)

Pull in the answers too.

Resolved tickets usually hold the answer, written by an agent who already solved it. That is the core of KCS.

No single signal is enough.

Each one is noisy. What matters is the same question showing up in several of them.

CHECKLIST

- [ ] Connect search logs, including no-result and no-click searches
- [ ] Turn on article feedback and capture the comment, not just the vote
- [ ] Link help center visitors to ticket requesters
- [ ] Pull bot transcripts, flagged resolved, escalated or abandoned
- [ ] Include solved tickets and agent replies

STEP 02 · DIAGNOSE

Diagnose the gap.

Turn thousands of scattered signals into a short, ranked list of named gaps. Three moves.

A **Cluster by intent**	Group signals by the question the customer was trying to answer, in their words. “How do I add a guest?” is actionable; “Access Management” is not. Keep the verbatims attached.
B **Size by customers**	Rank by distinct customers affected, not ticket count. Weight by revenue or plan tier, and watch direction: a small cluster growing 40% a week can matter more than a large flat one.
C **Label each cluster**	Exactly one label per cluster. The label decides the fix.

LABEL	THE TEST	EXAMPLE
● **Missing**	No published article answers the question	How to export audit logs; nothing covers it
● **Outdated**	An article contradicts the current product, pricing or UI	Shows a settings menu that moved last release
● **Unclear**	Correct, but customers still contact you, or it uses terms they don’t search	Says “provision seats”; customers search “add teammate”
● **Not a docs problem**	Caused by a bug, UX defect or policy. No article can fix it.	A sync error, or a disliked refund policy

Check coverage before you call it Missing. Many gaps are covered by an article customers cannot find, which makes them Unclear.

STEP 03 · FIX

Fix with one move per gap.

Each label maps to one move and one owner. Mixing them up is how teams rewrite articles that were never the problem.

LABEL	MOVE	HOW	APPROVER
Missing	Write it	Draft from agents' best resolved replies, not a blank page. AI drafts; a human approves.	Content owner for the area
Outdated	Update it	Tie articles to product areas. When release notes touch an area, diff the article against it.	Content owner, with a product check
Unclear	Rewrite it	Title it with the customer's question, add their synonyms, put the answer first.	Content owner
Not a docs problem	Route it	File in Jira or Linear with customer count, revenue at stake and verbatims.	The PM or engineering owner

Guardrails for AI-assisted fixes.

SET BEFORE IT RUNS

01 **Never invent product facts.** No made-up UI labels, menu paths, prices or limits.

02 **Nothing publishes without an approver.** Low-confidence drafts go to a human queue.

03 **Cap the volume per run.** A few good drafts beat fifty nobody reviews.

04 **Route defects, don't document them.**

05 **Keep a decision log.** What was proposed, approved, rejected and why. This is the loop's memory.

STEP 04 · VERIFY

Verify the topic, not the article.

A fix worked when fewer customers need help with that question, not when the article got more views. Measure every contact, search and chat in the cluster, 30 days before vs 30 days after.

METRIC	SHOULD	WATCH OUT FOR
Contacts per topic	Fall	Seasonality and growth: compare per 1,000 active customers
Search success	Rise	Use tickets created after search as the denominator (Zendesk)
Repeat contacts	Fall	Needs customer identity joined across channels
AI agent resolution on the topic	Rise	Many tools record an outcome on only some conversations

THE HEADLINE NUMBER

$$\Delta \text{ contact rate} = \frac{\text{contacts}_{\text{after}} / \text{active customers}_{\text{after}} - \text{contacts}_{\text{before}} / \text{active customers}_{\text{before}}}{\text{contacts}_{\text{before}} / \text{active customers}_{\text{before}}}$$

NO MOVEMENT IN 30 DAYS? BACK TO STEP 01 →

Reopen the cluster with its history attached. Often the first fix had the wrong label, like an “Unclear” article that was really a bug.

GETTING STARTED

The first 90 days.

Start small and manual, prove one topic moves, then automate. A loop that closes on one product area beats a dashboard covering everything.

DAYS 1-30

See the gaps.

- [ ] Connect the four signals for one product area
- [ ] Bring them together with tickets and chats in one place
- [ ] Rank the top 20 clusters by distinct customers
- [ ] Label each one
- [ ] Record a 30-day baseline per cluster

DAYS 31-60

Fix and measure.

- [ ] Name a reviewer and book a weekly 30-minute review
- [ ] Ship fixes for the top five clusters
- [ ] Route Not a docs problem clusters to Jira or Linear
- [ ] Start a decision log

DAYS 61-90

Close the loop.

- [ ] Compare each fixed topic against its baseline
- [ ] Send topics that did not move back to Detect
- [ ] Automate detection, with guardrails and caps
- [ ] Add a second area only once the first moves

WHAT TENDS TO GO WRONG

- Measuring article views instead of topic contacts
- Papering over bugs with articles
- Labeling every cluster “Missing”
- Automating detection with no reviewer time

Prove one topic moves. **Then automate.**

THE LOOP AT A GLANCE

Detect
Four signals, one place.

Diagnose
Cluster, size, label.

Fix
One move per label.

Verify
Topic contacts, 30 days.

NEXT STEP

See the loop on your own help center.

Enterpret can show you how this works on your own feedback data.

BOOK A DEMO →

enterpret.com/demo

SOURCES

01 Gartner: Only 14% of customer service issues are fully resolved in self-service (Aug 2024; 5,728 customers)

02 CX Dive: Self-service often falls flat. Here's how CX leaders can fix it (Aug 2024)

03 Gartner: Service leaders survey on conversational GenAI (Dec 2024; 187 leaders)

04 Gartner: More than half of service journeys start on third-party platforms (Jul 2025)

05 Consortium for Service Innovation: KCS v6 Practices Guide

06 Zendesk: Analyzing help center search results

07 Intercom: Fin AI Agent unresolved questions and content gaps