[PRODUCT NAME] Help Center

Information Architecture Reorganization

Optimized for customers and [AI ASSISTANT NAME] • [MONTH YEAR]

Executive Summary

The current **support.[PRODUCT NAME]** site has **[#]** top-level collections containing approximately **[#] articles** and **[#] total collections**. This proposal consolidates the site into **[#] top-level collections** organized around user tasks rather than internal product structure. The reorganization simultaneously optimizes for **[AI ASSISTANT NAME]** retrieval by eliminating the structural patterns that degrade AI answer quality: ambiguous collection names, duplicated concepts across articles, and an overloaded integration catalog that flattens **[#] sub-collections** into one bucket.

Two Audiences, One Structure

Every structural decision in this proposal is evaluated against two criteria: **Does this help a human find what they need?** and **Does this help [AI ASSISTANT] retrieve the right content and synthesize an accurate answer?** Where those goals conflict, the proposal calls it out. In most cases they align naturally, since [AI ASSISTANT] performs best when the same content principles that help human readers are in place.

What [AI ASSISTANT] needs from your content

[AI ASSISTANT] resolves customer questions by searching your help center, retrieving relevant passages, and generating answers with citations. Its accuracy depends on several structural factors in the underlying content.

[AI Assistant] Requirement	What It Means in Practice	Current Gap
One topic per article	Each article should answer one clear question or cover one discrete task. If the AI retrieves an article covering multiple topics, it may cite the wrong passage or hallucinate connections between sections.	[Example: Several collections bundle multiple concepts — e.g., Getting Started covers onboarding, navigation, and account access in overlapping articles.]
Unambiguous titles	Article and collection titles should match the language customers use when asking questions. The AI uses titles as strong retrieval signals.	[Example: A collection named ambiguously sounds like a different feature. Note any title

		that could be misread as covering a different topic.]
No duplication	When the same concept appears in multiple articles with slightly different wording, the AI may retrieve conflicting passages and either pick the wrong one or hedge its answer.	[Example: A "Getting Started" sub-collection overlaps with a top-level "Basic Concepts" collection. Audience-blended content in one section duplicates explanations found elsewhere.]
Consistent article structure	The AI chunks articles for retrieval. Consistent heading patterns, intro sentences, and section breaks help it identify where the relevant information lives within an article.	[Not assessed at the article level in this proposal, but structural inconsistency across collections suggests this is worth auditing.]
Logical collection boundaries	Collections act as context boundaries. When the AI retrieves from a well-scoped collection, it has stronger signals about relevance. An overloaded collection dilutes those signals.	[Example: An integration catalog contains every sub-integration as direct children with no categorical grouping.]

Current Site Structure

The homepage currently surfaces **[#]** top-level collections. Below is how they map to user needs, with notes on structural issues.

Homepage collections (current state)

Collection	Apparent Purpose	Issue
[Collection 1 — e.g., Getting Started]	Onboarding	[Overlaps with another collection; unclear where onboarding truly begins.]
[Collection 2 — e.g., Basic Concepts]	Foundational knowledge	[Overlaps with Getting Started and a nested sub-collection elsewhere.]
[Collection 3]	[Proprietary concept] onboarding	[Unclear to new users unfamiliar with internal terminology; overlaps with onboarding.]
[Collection 4]	Login / account access	[Narrow topic that could nest under a broader admin section.]
[Collection 5]	Platform UI orientation	[Overlaps with Collection 6.]
[Collection 6]	Dashboard UI	[Could consolidate with Collection 5.]
[Collection 7]	Admin: users and roles	Well-scoped.

[Collection 8]	Data domains ([#] sub-collections)	Well-structured internally; sub-collections are logical.
[Collection 9]	Formula / expression reference	[Closely related to data management but isolated at top level.]
[Collection 10 — Integration Catalog]	Integration catalog ([#] sub-collections)	[Massively overloaded; ambiguous name; all integrations and mapping collections are flat siblings.]
[Collection 11]	Managing subscriptions / plans	[Name collision with Collection 10 creates confusion for both humans and the AI assistant.]
[Collection 12]	Observability / monitoring	[Could group with error troubleshooting for a complete ops workflow.]
[Collection 13]	Error diagnosis ([#] sub-collections)	Well-structured internally; separated from monitoring.
[Collection 14 — API]	API reference (with Examples sub-collection)	[Audience-blended content: subscribers + partners/developers.]
[Collection 15]	Partner / developer guide ([#] sub-collections)	Well-structured internally; sub-collection progression is logical.
[Collection 16]	Implementation guidance	[Ambiguous positioning relative to Collection 15.]
[Collection 17]	Platform settings	[Could consolidate into a platform admin section.]

Proposed Structure

The reorganization reduces **[#]** top-level collections to **[#]**, each mapped to a distinct user journey. Below, each proposed collection is described with its purpose, what it absorbs from the current structure, and specific [AI ASSISTANT] optimization notes.

1. [Proposed Collection 1 — e.g., Getting Started]

Absorbs: [List the current collections being merged here — e.g., Getting Started, Basic Concepts, [Proprietary Concept] Experience, Account Access]

Purpose: A single guided entry point for new users. The current multi-collection onboarding sprawl means [AI ASSISTANT] may retrieve answers from any of them for the same question, potentially citing inconsistent explanations. Consolidating into one collection with clearly scoped articles gives [AI ASSISTANT] a single authoritative source for each onboarding concept.

Suggested sub-structure: [What Is [PRODUCT NAME]] (platform overview and key terminology), [Logging In and Account Access] (consolidate account access content here), [Navigating the Platform] (light orientation to the UI, linking out to the Platform Guide for depth),

[Your First Integration or Workflow] (a guided walkthrough that links to the relevant technical section for detail).

[AI ASSISTANT] optimization: Each article in this collection should open with a one-sentence summary that directly answers the most common form of the question it addresses. For example, an article titled "What is a [Key Concept]?" should begin with a crisp definition before expanding. This gives [AI ASSISTANT] a high-confidence passage to cite without reading deep into the article.

2. [Proposed Collection 2 — e.g., Platform Guide]

Absorbs: [Navigation, Dashboard, User/Role Management, Settings, Subscription Management — list the current collections being merged]

Purpose: Everything about using the [PRODUCT NAME] platform day-to-day. This eliminates the current split between UI and settings sections as separate top-level entries, which forces users (and [AI ASSISTANT]) to guess which collection holds the answer to a platform question.

Suggested sub-collections: [Navigation & Dashboard] (merge the separate navigation and dashboard collections), [User & Role Management] (keep as-is if well-scoped), [Settings], [Managing Subscriptions] (rename if there is a name collision with an integration catalog).

[AI ASSISTANT] optimization: Renaming any ambiguously named collection resolves the most common retrieval errors. A user asking [AI ASSISTANT] about managing their plan should not retrieve content from an integration catalog. Distinct, descriptive names resolve this.

3. [Proposed Collection 3 — e.g., Data & Transformations]

Absorbs: [Data Management and sub-collections, Dynamic Formulas or equivalent — list current collections]

Purpose: A unified home for understanding how data is structured, moved, and transformed within [PRODUCT NAME]. Any formula or expression reference that floats as a disconnected top-level collection should be absorbed here.

Suggested sub-collections: Keep the existing data-domain sub-collections and add the formula/expression reference as a sibling sub-collection within this grouping.

[AI ASSISTANT] optimization: This grouping gives [AI ASSISTANT] better context boundaries. When a user asks about formula syntax, [AI ASSISTANT] retrieves from a collection explicitly about data transformation rather than from an orphaned top-level collection with no contextual neighbors.

4. [Proposed Collection 4 — e.g., Integration Catalog]

Absorbs: [Current integration catalog collection and all [#] sub-collections, all integration-specific and mapping sub-collections]

Purpose: [Replace the overloaded collection name with a purpose-built, categorized integration library. This is typically the highest-impact change in a reorganization like this. When all integrations and mapping collections sit as flat siblings under one parent, every integration retrieval query competes against all peers with no categorical signal to narrow results.]

Key structural change — nest mapping collections: Move mapping or configuration collections inside their parent integration collection. A user (or [AI ASSISTANT]) navigating to "[Integration Name]" should find everything in one place: installation, configuration, and mappings.

Suggested category sub-collections: Group integrations by system type to make the catalog browsable and to give [AI ASSISTANT] categorical context.

Category	Integrations
[Category A — e.g., eCommerce]	[Integration 1], [Integration 2], [Integration 3], …
[Category B — e.g., ERP & Accounting]	[Integration 1], [Integration 2], …
[Category C — e.g., Point of Sale]	[Integration 1], [Integration 2], …
[Category D — e.g., CRM & Marketing]	[Integration 1], [Integration 2], …
[Category E — e.g., PIM & Product Data]	[Integration 1], [Integration 2], …
[Category F — e.g., Shipping & Fulfillment]	[Integration 1], [Integration 2], …
[Category G — e.g., HR & Workforce]	[Integration 1], [Integration 2], …
[Category H — e.g., Communication]	[Integration 1], [Integration 2], …
[Add more categories as needed]	…

[AI ASSISTANT] optimization: Category grouping dramatically improves retrieval precision. When a user asks "How do I connect my [system type]?", [AI ASSISTANT] can surface integrations from the relevant category rather than scanning all sub-collections. Nested mapping docs also mean [AI ASSISTANT] can answer mapping-specific questions with the full integration context available, reducing hallucination risk from context gaps.

Article-level guidance for [AI ASSISTANT]: Each integration collection should follow a consistent article template: Installation Instructions, Connection Configuration, Mapping Overview (linking to or containing mapping docs), and Troubleshooting. Consistent titles across integrations ("[Integration] Installation Instructions") mean [AI ASSISTANT] can pattern-match user questions like "How do I install [X]?" to the right article with high confidence.

5. [Proposed Collection 5 — e.g., Building Integrations / Partner Guide]

Absorbs: [Partner/developer guide collection and its sub-collections, Implementation & Maintenance, Best Practices — list current collections]

Purpose: A unified home for partners and developers building or implementing integrations on the [PRODUCT NAME] platform. The current structure often splits this audience across multiple top-level collections with overlapping content.

Keep the existing sub-collection progression: If there is a logical implementation lifecycle in your current sub-collections (e.g., Core Concepts → Planning → Connectivity → Configuration → Webhooks → Optimization → Testing & Deployment), absorb any stray "Implementing & Maintaining" or "Best Practices" collections into the appropriate stages rather than keeping them as separate entries.

[AI ASSISTANT] optimization: Rename any sub-collection whose title collides with a top-level collection name. Title collisions are one of the most common causes of [AI ASSISTANT] retrieving content from the wrong section.

6. [Proposed Collection 6 — e.g., API Reference]

Absorbs: [API collection, Examples sub-collection — list current collections]

Purpose: Technical API documentation and code examples. Keep this as a distinct section, but clarify the audience. Separate articles by audience: subscriber-facing API usage vs. partner-facing endpoint reference.

[AI ASSISTANT] optimization: Articles in this collection should use precise, technical titles that match how users ask API questions (e.g., "How to authenticate API requests" rather than "Authentication Overview"). Code examples should appear in clearly labeled sections so [AI ASSISTANT] can cite them without pulling in surrounding narrative.

7. [Proposed Collection 7 — e.g., Monitoring & Troubleshooting]

Absorbs: [Integration Monitoring & Diagnostics, Error Troubleshooting, product-specific error collections — list current collections]

Purpose: Unify the observe-diagnose-fix workflow into one section. Currently, a user whose integration is failing may have to check a monitoring collection and then switch to a separate error troubleshooting collection. Grouping them means [AI ASSISTANT] can retrieve monitoring context and error resolution from the same collection, producing more complete answers.

Suggested sub-collections: [Integration Monitor] (how to use the monitoring tools), [[PRODUCT NAME] Errors] (keep as-is), [External System Errors] (rename from any "Errors Outside of [PRODUCT NAME]" phrasing for clarity), [Diagnostics & Resolution Guides] (consolidate cross-cutting troubleshooting procedures).

[AI ASSISTANT] optimization: Error articles are some of the highest-value content for [AI ASSISTANT] because error messages are highly specific queries. Each error article should lead with the exact error message text or error code in the title or first sentence. This gives [AI ASSISTANT] an exact-match retrieval signal.

Before and After: Top-Level Navigation

Current ([#] collections)	Proposed ([#] collections)
[Collection 1]	[Proposed Collection 1 — e.g., Getting Started]
[Collection 2]	(merged into [Proposed Collection 1])
[Collection 3]	(merged into [Proposed Collection 1])
[Collection 4]	(merged into [Proposed Collection 1])
[Collection 5]	[Proposed Collection 2 — e.g., Platform Guide]
[Collection 6]	(merged into [Proposed Collection 2])
[Collection 7]	(sub-collection of [Proposed Collection 2])
[Collection 8]	[Proposed Collection 3 — e.g., Data & Transformations]
[Collection 9]	(merged into [Proposed Collection 3])
[Collection 10] ([#] sub-collections)	[Proposed Collection 4 — e.g., Integration Catalog (categorized)]
[Collection 11]	[Proposed Collection 5 — e.g., Building Integrations]
[Collection 12]	(merged into [Proposed Collection 5])
[Collection 13]	[Proposed Collection 6 — e.g., API Reference]
[Collection 14]	[Proposed Collection 7 — e.g., Monitoring & Troubleshooting]
[Collection 15]	(merged into [Proposed Collection 7])
Add rows as needed …	…

[AI ASSISTANT] Optimization: Cross-Cutting Recommendations

Beyond the structural reorganization, the following article-level practices will materially improve [AI ASSISTANT]'s answer quality across the entire site.

1. Lead every article with a direct answer

The first one or two sentences of every article should directly answer the question the article addresses, before expanding into detail. [AI ASSISTANT] heavily weights early content in an article when generating answers. An article that buries the answer after three paragraphs of context forces [AI ASSISTANT] to extract from deeper in the text, increasing the risk of an imprecise or incomplete citation.

2. Use consistent, natural-language titles

Titles should match the way users phrase questions. "How to Configure [Integration] Connection Settings" is better than "[Integration] Configuration" because it matches a question pattern. Avoid internal jargon in titles — terms that are meaningful inside your organization may mean nothing to a new user typing a question into a chat widget.

3. Deduplicate aggressively

Audit for cases where the same concept is explained in multiple articles. When [AI ASSISTANT] retrieves two articles with slightly different explanations of the same thing, it may either pick the less accurate one or hedge with a vague answer that cites both. Onboarding collections and any audience-blended sections (such as an API collection) are typically the highest-priority deduplication targets.

4. Standardize integration article templates

Every integration should follow the same article structure: Installation Instructions, Connection Configuration, Mapping Overview, and Troubleshooting. Consistent titles and heading patterns let [AI ASSISTANT] pattern-match across integrations, so a question like "How do I install [any integration]?" reliably retrieves the right article.

5. Front-load error messages in troubleshooting articles

When a user pastes an error message into chat, [AI ASSISTANT] performs essentially an exact-match search. If the error text appears in the article title or first sentence, retrieval is nearly perfect. If it is buried in paragraph four, [AI ASSISTANT] may miss it entirely. Structure error articles as: title containing the error, first sentence restating the error and its cause, then resolution steps.

6. Avoid multi-topic articles

An article that covers three loosely related features gives [AI ASSISTANT] a low-confidence retrieval signal for any one of them. Split multi-topic articles into focused single-topic articles. Each article should pass the test: "Can I describe what this article is about in one sentence?"

7. Add a brief description to every collection

Your help center platform likely allows collection descriptions. These serve as context signals for [AI ASSISTANT] and as orientation for human readers. A description like "Step-by-step guides for connecting external systems to [PRODUCT NAME], covering authentication, data mapping, and webhook configuration" gives [AI ASSISTANT] explicit scope boundaries.

Implementation Approach

This reorganization can be executed incrementally without taking the help center offline. The recommended sequence prioritizes changes with the highest impact on both user experience and [AI ASSISTANT] accuracy.

Phase 1: Rename and consolidate (low effort, high impact)

Rename the most ambiguously named collections first — particularly any that share a name with another collection or use internal jargon. These changes resolve the most damaging naming ambiguities immediately without requiring content moves.

Phase 2: Merge onboarding collections

Consolidate the onboarding and foundational knowledge collections into a single Getting Started collection. Deduplicate overlapping articles during the merge. Redirect old URLs.

Phase 3: Restructure the integration catalog

Create category sub-collections under the integration catalog. Nest mapping collections inside their parent integration. This is typically the most labor-intensive phase, but has the largest payoff for [AI ASSISTANT] retrieval quality and user navigation.

Phase 4: Consolidate platform and operations collections

Merge navigation, dashboard, settings, and subscription management sections into a unified Platform Guide. Merge monitoring and error troubleshooting sections into a unified Monitoring & Troubleshooting section. Absorb any stray implementation or best-practices collections into the partner/developer guide.

Phase 5: Article-level [AI ASSISTANT] optimization

Audit and rewrite article titles for natural-language question matching. Add lead-sentence summaries to all articles. Standardize integration article templates. Deduplicate cross-collection content. This phase is ongoing and can run in parallel with the structural work.