ATLORIA

Turning Analytics into Documentation Improvement Actions

August 25, 2026

Reviewing the analytics signals that should trigger documentation changes

In Atloria, start from the **Analytics & Insights** area in the admin workspace when it is available to your team, and combine that with the project-level analytics views you already use in Reviewing Documentation Performance Across Projects. The goal here is not to look at every number. It is to find the signals that clearly point to a documentation problem you can fix.

Review page-level reports first. Focus on pages that show a mismatch between attention and results, such as:

- high page views with weak engagement
- pages where readers leave quickly
- pages that attract traffic but do not help readers continue to the next step
- articles tied to repeated support questions

Compare the main page signals together instead of reading them one by one:

- **Views** tell you whether people are reaching the page
- **Bounce rate** helps you spot pages readers leave without exploring further
- **Time on page** helps you judge whether readers are skimming, struggling, or finishing quickly
- **Task completion indicators** help you see whether the article supports the intended action

Then move to search reporting. The most useful fields are:

- **Top queries**
- **Zero-result searches**
- **Query-to-click rate**

These help you spot missing terms, unclear page titles, weak headings, or missing troubleshooting content. For example, if readers search for an upgrade error using one phrase but your article uses different wording, the page may exist but still stay hard to find.

Before deciding what to change, cross-check the analytics with nearby evidence:

- support ticket categories
- release issue patterns
- repeated escalation themes from support agents
- version-related confusion after a release

[SCREENSHOT: Analytics report showing page views, exits, search queries, and support-related trends side by side]

When the same issue appears in page analytics, search behavior, and support activity, that is usually the strongest signal that the documentation needs an update.

Classifying findings into content, audience, release, and support actions

Once you have a short list of problem pages, sort each finding into the type of action it needs. In Atloria, this step helps you avoid making the wrong fix. A page with high traffic and low engagement does not always need more text. Sometimes it needs clearer audience targeting, version guidance, or better support links.

Use **content actions** when readers are reaching the page but not getting through it successfully. These are the most common fixes for pages with strong traffic and weak engagement. Typical content actions include:

- rewriting the opening paragraph so the purpose is clear immediately
- restructuring headings so the steps are easier to scan
- adding missing setup steps or prerequisites
- moving troubleshooting details closer to the step where the problem happens

Use **audience actions** when one group succeeds and another struggles. If a page performs well for administrators but poorly for end users, or if support agents use a page that was written for customers, the issue is often targeting rather than quality. In that case, improve:

- navigation labels
- role-specific entry points

- audience labels on landing pages
- separate paths for different reader types

Use **release actions** when analytics show version confusion. Common signs include:

- spikes in searches right after launch
- repeated visits to upgrade pages
- readers landing on outdated instructions
- support volume increasing after a release

These findings usually point to missing release notes, weak version banners, unclear compatibility guidance, or missing known issues content.

Use **support actions** when the same issue keeps appearing in escalations or long support conversations. That usually means the internal knowledge base needs:

- symptom-based troubleshooting
- decision trees
- issue-specific notes
- direct links from support workflows to the right article section

A simple way to sort findings is with a tracker like this:

Finding	Best action type	Typical update
High traffic, low engagement	Content	Rewrite intro, reorder steps, add missing details
Good results for one role, poor results for another	Audience	Split content path, relabel navigation, add role cues
Post-release confusion or version spikes	Release	Add version banner, release notes, known issues
Repeated escalations or long support handling	Support	Add troubleshooting flows and support-facing notes

Turning page analytics into concrete content updates

When a page clearly needs a content fix, open the page performance view in Atloria and choose one article to work on first. Pick a page with a strong signal, such as high views combined with low completion, or a page where readers often leave after the same section. This keeps your work focused on pages that can produce visible improvement.

1. Open the page performance report and select the target article. Look for pages with high traffic, high exits, or weak completion signals. If several pages qualify, start with the one tied to the most important workflow, such as setup, publishing, approvals, or version release.
2. Review the article structure against the behavior signals. Check the page title, the main heading, the table of contents, callout blocks, screenshots, and the order of the steps. If readers drop off early, the opening may be too vague or missing prerequisites. If they leave midway, the problem is often a missing step, unclear instruction, or a gap between headings and what readers searched for.
3. Match the update to the signal:
    - Add prerequisites when readers exit before starting the task
    - Add screenshots when readers fail at a specific step repeatedly
    - Expand troubleshooting when the page is linked to support-heavy issues
    - Rename headings when search terms do not match the words on the page
    - Break long sections into shorter procedures when readers stop halfway through
4. Record the planned update in your editorial tracker so the work does not disappear after the analytics review.

Use a tracker with fields like these:

Field	What to record
Page	The article you are updating
Issue type	Low completion, high exits, search mismatch, support-heavy topic
Proposed update	What you will change on the page
Owner	Who is making the update
Review date	When you will check results again

[SCREENSHOT: Page analytics view beside an editorial tracker with issue type, owner, and review date]

This approach turns a vague result like “this page underperforms” into a specific edit you can publish and measure.

Adjusting audience targeting when the wrong readers reach the page

Sometimes the content is accurate, but the wrong people are landing on it. In Atloria, you can usually spot this by comparing referral sources, entry pages, and any available role or persona filters in your analytics views. If a page meant for administrators is attracting end users, or a support-focused troubleshooting article is being used as customer-facing guidance, the fix should focus on audience targeting.

1. Confirm who is reaching the page. Check the entry page, referral source, and any audience or role filters available in your analytics reports. A page that performs poorly may simply be attracting readers who should have started somewhere else.
2. Make the intended audience obvious before the article opens. Update navigation labels, landing page copy, and page titles so readers can tell whether the content is for administrators, end users, or support teams. If the label is too broad, readers often click into the wrong article and leave quickly.
3. Split mixed-purpose content when one page serves different roles. For example, if one article combines customer setup instructions with internal diagnostic steps, separate those into distinct pages or clearly separated sections. Readers should not have to scan past unrelated guidance to find the path that applies to them.
4. Add audience cues inside the article itself. Use clear labels such as:
    - **Who should use this**
    - **Prerequisites**
    - **Permissions required**
    - **Use this procedure if**

These cues help readers decide quickly whether they are in the right place.

[SCREENSHOT: Documentation page showing role-specific labels, prerequisites, and separate navigation entries]

Audience targeting updates are especially useful when a page gets steady traffic but inconsistent outcomes. If one reader group succeeds and another does not, changing the wording, labels, and entry points often works better than rewriting the full article. If you need a broader approach to audience structure, use this together with Applying Audiences to Documentation Structure and Content Decisions.

Using analytics to improve release readiness and support coverage

Analytics become especially valuable around releases, because they show where readers hesitate, search repeatedly, or return to the same guidance after launch. In Atloria, compare pre-release and post-release patterns to see whether your documentation prepared readers properly or left gaps that support teams now have to fill.

1. Review version-specific search activity and traffic spikes. Look for sudden increases in visits to upgrade instructions, compatibility topics, migration steps, and known limitation pages. Repeated searches around the same release term usually mean readers cannot find the right version guidance quickly enough.
2. Check whether readers are revisiting the same release-related pages several times. That often signals uncertainty rather than success. If readers keep returning to upgrade content, they may need clearer release notes, a checklist, or a better explanation of what changed.
3. Create or update release-readiness content based on what the analytics show. The most common assets to improve are:
    - release notes
    - upgrade checklists
    - compatibility tables
    - known issues pages
    - version banners on affected articles
4. Compare support activity with documentation usage. If support tickets, escalation tags, or agent conversations cluster around the same issue, check whether the internal knowledge base already covers it. If it does not, add support-facing content that helps agents move faster and answer consistently.

Prioritize support updates such as:

- symptom-based troubleshooting pages
- issue triage flows
- links from support macros to exact article sections
- internal notes for recurring release problems

[SCREENSHOT: Release-related analytics showing traffic spikes alongside support issue trends]

This is where analytics stop being just reporting and start guiding operational work. If a release creates confusion, the right response is not only to update the public

documentation, but also to strengthen the internal support material that agents rely on during the same period. For related release workflows, connect this work with Preparing a Version for Final Release Review.

Validating that your documentation changes improved the right outcomes

After you publish updates in Atloria, return to the same reporting window and measure the exact outcomes you were trying to improve. Do not switch to a different report or a different time frame unless you have to. The cleanest comparison comes from checking the same page, the same search terms, and the same support topic before and after the change.

1. Reopen the original page and search reports. Check whether the updated article shows better search success, fewer exits, stronger task completion signals, or lower support demand. The metric you watch should match the problem you were fixing.
2. Watch for misleading improvements. A traffic increase alone does not prove the page is better. Release-week spikes, seasonal usage, and URL changes can all distort the numbers. If a page moved or was renamed, make sure you are comparing the correct article history.
3. If traffic improves but outcomes do not, revisit the diagnosis. The original issue may not have been content clarity. It may have been:
    - an audience mismatch
    - release confusion
    - weak troubleshooting depth
    - poor navigation into the article
4. Add the page back into a recurring review cycle if the result is still unresolved. Set an owner, a review date, and a threshold that tells your team when the page should return to the analytics queue.

A simple review rhythm might include:

- weekly checks for release-related pages
- monthly checks for high-traffic evergreen content
- scheduled follow-up after major edits
- shared ownership between documentation and support leads

[SCREENSHOT: Before-and-after analytics comparison for an updated documentation page]

If the numbers improve in the right place, keep the change and move to the next priority. If not, refine the action and test again. That same habit of checking outcomes against the original problem is what turns analytics into a repeatable documentation improvement process.

Overview

Atloria gives you several places to spot documentation problems, but the real value comes from turning those signals into specific actions. This guide focuses on that last step: deciding what to change after analytics reveal weak content performance, audience confusion, release friction, or support knowledge gaps.

Use this workflow when you already know how to review project and cross-project reporting and need to decide what to do next. Instead of treating analytics as a dashboard you glance at occasionally, use it as a working list for documentation updates. In practice, that means identifying a problem page or topic, classifying the issue, making the right type of change, and then checking whether the result improved.

This guide covers how to:

- identify the analytics signals that deserve action
- sort findings into content, audience, release, or support work
- turn page-level metrics into concrete edits
- adjust navigation and labels when the wrong readers reach a page
- improve release-readiness content and support-facing knowledge
- validate whether your updates actually solved the problem

You do not need every report in Atloria to use this process well. A smaller set of reliable signals is usually enough:

- page performance trends
- search behavior
- version-related traffic changes
- support-linked issue patterns

If you need help reading those reports first, go back to Reviewing Documentation Performance Across Projects and Using Analytics to Prioritize Documentation

Improvements. Those guides explain how to identify patterns; this one shows how to turn those patterns into editorial, audience, release, and support actions your team can assign and track.

Prerequisites

Before you use this workflow in Atloria, make sure you have access to the reporting and content areas needed to both review the issue and act on it. You do not need every admin feature, but you do need enough visibility to connect analytics with the pages, versions, and support topics involved.

You should already be able to:

- open the relevant **Analytics** views for your project or admin workspace
- review documentation pages and their structure
- identify the affected version, release, or audience path
- update content directly or hand changes to the person who owns the page
- compare documentation trends with support or release patterns used by your team

It also helps if you already have:

- an editorial tracker or shared work list
- a clear page owner for major documentation areas
- a release review process for version-specific content
- a way to capture recurring support issues

Have these inputs ready before you start:

- the page or topic you want to review
- the date range you are comparing
- the key metric that triggered concern
- any related search terms, support themes, or release notes
- the team member responsible for the update

This guide works best after you have already reviewed broader performance patterns in Atloria. If you have not done that yet, start with:

- Analyzing Documentation Performance Across Projects
- Using Analytics Reporting Across Enterprise and Project Views
- Reviewing Documentation Performance Across Projects

From there, you can move directly into action planning: choosing the right fix, publishing the update, and checking whether the change improved the outcome you were targeting.