Understand Source Analysis Parsing

September 10, 2026

Understand Source Analysis Parsing

Source analysis parsing prepares your Atloria project for product intelligence and user documentation. It examines the project source, organizes what it finds into meaningful categories, and identifies how application elements relate to one another.

Overview

When you add or maintain a project in Atloria, source analysis helps turn a collection of source files into understandable product context. Rather than treating every file as isolated, the parsing process identifies application elements such as components, pages, routes, services, repositories, hooks, stores, controllers, and middleware.

The parser also organizes findings by category. These categories include **FRONTEND**, **BACKEND**, **DATABASE**, **API SCHEMA**, **CONFIG**, **DOCUMENTATION**, and **INFRASTRUCTURE**. This organization helps you understand which parts of a project have been analyzed and makes it easier to focus on the area most relevant to your documentation or product-intelligence work.

In addition to identifying elements, source analysis records relationships between them. For example, a component may **RENDERS** another component, **USES HOOK**, **USES CONTEXT**, **CALLS API**, or **USES SERVICE**. These relationships provide the context needed to explain how a user-facing workflow is supported across different areas of your project.

> 📷 *Screenshot pending: Project Source area showing source analysis options and parsed project context.*

Prerequisites

Before you begin source analysis parsing, make sure you have the following:

- Access to the appropriate Atloria project from **Projects** or **1. Hub: All Projects**.

- A project whose source is available for analysis.
- Permission to view project source and configuration information.
- A clear understanding of the project area you want to investigate, such as frontend behavior, backend services, database access, or API behavior.
- Access to **Source** and, when configuration changes are needed, **2. Parser Settings**.
- A reason to refresh or review analysis, such as newly added features, changed integrations, updated API behavior, or documentation that needs current context.

Step-by-Step Instructions

1. **Open the project from the project hub.**

   Start from **Projects** or select **1. Hub: All Projects**. Use **Search projects...** if you need to locate a specific project quickly.

   Select the project you want to analyze. If you are returning from another project area, use ← **Back to Projects** or ← **Back to Projects** to return to the project list.

2. **Go to the project's source area.**

   In the selected project, choose **Source**. This is the area where you review the project's source-based context and prepare it for product intelligence and documentation work.

   Source analysis is especially useful before you create or revise content in **User Guides**, because it provides evidence about the application structure and the connections between features.

3. **Review parser configuration when needed.**

   Select **2. Parser Settings** when you need to review the settings that control how source analysis is prepared for the project.

   Use this area before relying on a new or refreshed analysis result. Parser settings matter because they determine the context Atloria uses when organizing source findings. If your project has changed substantially—for example, if it now includes a new API, database layer, or frontend application—review the settings before continuing.

   > 📷 *Screenshot pending: Parser Settings screen for reviewing source analysis preparation for a project.*

4. **Run or review source analysis for the project.**

   Start the project's source analysis from the available source-analysis controls in the **Source** area. Atloria coordinates the parsing work across the project rather than limiting the review to one type of source.

   As parsing proceeds, Atloria classifies findings into categories such as:

   - **FRONTEND** for user-facing application elements
   - **BACKEND** for server-side capabilities
   - **DATABASE** for data access and data-model-related elements
   - **API SCHEMA** for API definitions and contracts
   - **CONFIG** for configuration-related information
   - **DOCUMENTATION** for documentation-related source context
   - **INFRASTRUCTURE** for deployment and supporting-platform context

   These categories help you interpret the scope of the analysis. For example, a change to a visible product screen may involve both **FRONTEND** findings and related **BACKEND** or **API SCHEMA** findings.

5. **Use entity findings to understand the application structure.**

   Review the entities identified by the analysis. Depending on your project, Atloria can recognize items such as **COMPONENT**, **HOOK**, **PAGE**, **LAYOUT**, **ROUTE**, **CONTEXT**, **PROVIDER**, **STORE**, **CONTROLLER**, **SERVICE**, **REPOSITORY**, and **MIDDLEWARE**.

   Treat these findings as a map of the product's structure. For user-facing documentation, start with pages, layouts, routes, and components. Then follow related hooks, contexts, stores, and services when you need to understand why a screen behaves in a particular way.

6. **Follow relationships to build feature context.**

   Review the relationships shown for parsed entities. Relationships can indicate that one item **RENDSERS** another item, **USES HOOK**, **USES CONTEXT**, **PROVIDES CONTEXT**, **USES STORE**, **DISPATCHES ACTION**, **CALLS API**, **IMPLEMENTS API**, **DEPENDS ON**, **USES SERVICE**, **INJECTS**, or **QUERIES MODEL**.

   Use these connections to trace a feature beyond the initial screen or page. For example, a visible component may render another component, use a hook for its behavior, and call an API through a service. This broader context is valuable when

you need to distinguish a simple display change from a change that affects data, permissions, or workflow behavior.

7. **Use the parsed context when creating user documentation.**

   After reviewing the source analysis, go to **User Guides** or 📋 **User Manuals** to create or update user-facing content. Use the analysis to confirm feature names, workflow dependencies, and the likely impact of a change.

   If you need to locate existing material, use **Search documents, categories, projects...**. Search helps you check whether related documentation already exists before you create a new guide.

Tips and Best Practices

- **Start with the user-facing layer.** For a manual or user guide, begin with **PAGE**, **LAYOUT**, and **COMPONENT** findings. Then use relationships such as **CALLS API** or **USES SERVICE** only when they help explain visible behavior.
- **Do not assume a frontend change is isolated.** A visible component can depend on a hook, context, store, service, API, or database query. Review its relationships before documenting what a change means for users.
- **Use categories to narrow your review.** If you are documenting a screen interaction, focus first on **FRONTEND**. If the interaction involves saved data or an integration, expand your review to **BACKEND**, **DATABASE**, and **API SCHEMA**.
- **Review parser settings after major project changes.** New application areas, services, APIs, or configuration can affect the completeness and usefulness of your source context.
- **Use analysis as evidence, not as user-facing wording.** Parsed entity and relationship names describe project structure. In user documentation, translate that structure into what users see and do on screen.
- **Keep related documentation aligned.** When source analysis reveals that one feature depends on another, check linked guides so that instructions remain consistent across the product.

Related Pages

- Manage Projects
- Configure Parser Settings

- Review Project Source
- Create User Guides
- Use Product Intelligence
- Search Projects and Documentation

Troubleshooting

I cannot find the project I need to analyze

Return to **Projects** or **1. Hub: All Projects** and use **Search projects...**. Confirm that you have access to the correct project before looking for **Source** or **2. Parser Settings**.

The analysis does not provide enough context for the feature I am documenting

Review related categories instead of looking only at the initial frontend finding. A user-facing feature may be connected to a service, API implementation, data query, or configuration item. Follow relationships such as **USES SERVICE**, **CALLS API**, and **QUERIES MODEL** to gather fuller context.

The project has changed, but the analysis no longer reflects the current structure

Open **2. Parser Settings** and review the project's analysis preparation. Then refresh or rerun source analysis from **Source** using the available controls. Recheck the affected categories and relationships before updating content in **User Guides**.