# Analyze a Connected Codebase

September 10, 2026

## Analyze a Connected Codebase

This page explains how to create a project from a connected source repository and use the resulting project workspace to understand its structure. Source analysis helps you turn a codebase into organized technical documentation that is easier to review, share, and maintain.

### Overview

When you connect a codebase to Atloria and create a project, Atloria analyzes the source to build an understanding of the application. The analysis identifies parts of the project such as frontend elements, backend services, database-related code, API schemas, configuration, documentation, and infrastructure.

You need this workflow when you are onboarding to an unfamiliar application, preparing technical documentation, or reviewing how parts of a project relate to each other. Rather than starting with individual files, you can begin from the project workspace and open the generated documentation through **Technical Docs**.

After a project is created, it becomes the active project in your workspace. From there, you can use project areas such as **Technical**, **Source**, versions, changes, analytics, and technical documentation to move between a high-level understanding and the supporting source material.

> 📷 *Screenshot pending: The Projects hub showing the New Project action and project search*

### Prerequisites

Before you start, make sure you have:

- Access to Atloria and permission to create or open projects.
- A source repository that you can connect to Atloria.

- Access to the repository provider you use, such as GitHub, GitLab, or Bitbucket.
- A repository or source URL ready to provide during project setup.
- A clear project name so you can find the project later in the **Projects** list.
- Enough time for Atloria to process the codebase before reviewing the generated technical documentation.

## Step-by-Step Instructions

1. **Open the Projects hub.**

   From your Atloria workspace, select **Projects**. The Projects hub is where you can view existing projects, switch your focus to a specific project, and begin creating a new one.

   If you already know the project name, use **Search projects...** to locate it. Use **Search across all projects...** when you need to search more broadly across the projects available to you.

2. **Start a new project.**

   Select **New Project**. Atloria opens the project-creation flow, where you provide the source location and project details.

   > 📷 *Screenshot pending: The New Project flow for connecting a repository source*

3. **Connect the source repository.**

   In the connection step, select the repository provider that hosts your codebase. Atloria supports repository connections represented by GitHub, GitLab, and Bitbucket options.

   Provide the repository source requested by the setup screen. Confirm that you are connecting the intended codebase, especially if your organization has similarly named repositories or multiple application versions.

4. **Create the project.**

   Enter the required project information, then select **Create project**. Atloria adds the repository as a project and begins building project understanding from the connected source.

During analysis, Atloria organizes discovered code into categories. These categories progress through the codebase in a meaningful order:

- **FRONTEND** for user-facing application code
- **BACKEND** for server-side functionality
- **DATABASE** for persistence-related code
- **API SCHEMA** for interfaces and data contracts
- **CONFIG** for project configuration
- **DOCUMENTATION** for existing documentation content
- **INFRASTRUCTURE** for deployment and environment-related resources

This organization matters because it helps you review a connected application as a system rather than as an unstructured collection of files.

5. **Open the focused project.**

   After the project is available, open it from **Projects** or select **Open Project Dashboard**. The project dashboard is your starting point for the active project.

   If you need to return to the project list, select ← **Back to Projects** or ← **Back to Projects**, depending on the screen you are viewing.

6. **Review the generated technical documentation.**

   In the focused project, select **Technical Docs**. This area presents the documentation built from the source analysis.

   Use the technical documentation to understand the important entities Atloria identifies in the project. Depending on the codebase, these can include components, hooks, pages, layouts, routes, contexts, providers, stores, controllers, services, repositories, and middleware.

   > 📷 *Screenshot pending: The Technical Docs area for a focused project, showing analyzed project documentation*

7. **Follow relationships from one part of the system to another.**

   Review how identified items relate to each other. For example, a component can render another component, or a component can use a hook. These relationships help you understand how a user-facing screen connects to the logic and services that support it.

Start with the area that matches your goal. For a screen behavior, begin with frontend entities. For a data or server behavior, review backend, database, and API schema entities. This approach keeps your investigation focused while still allowing you to trace connections across the codebase.

8. **Check the source when you need confirmation.**

   Select **View Source** when you need to verify how a documented item is represented in the connected repository. You can use the **Source** area as supporting evidence while using **Technical Docs** as the primary high-level view.

   When your review is complete, select **Export Documentation** if you need to share the generated documentation outside Atloria.

## Tips and Best Practices

- **Create one project for each meaningful codebase.** A focused project makes its documentation easier to navigate and prevents unrelated applications from being analyzed together.
- **Begin with Technical Docs, not individual source files.** The documentation provides the project-level context first. Use **View Source** only when you need deeper detail or want to validate a specific conclusion.
- **Review the project by category.** Start with **FRONTEND** when documenting user experiences, then follow connections to **BACKEND**, **DATABASE**, and **API SCHEMA** as needed. This mirrors how application behavior commonly moves from the interface to supporting services and data.
- **Use relationships to avoid isolated analysis.** A component that renders another component or uses a hook is part of a larger flow. Reviewing these connections gives you a more reliable understanding than reading one entity alone.
- **Use versions and changes for ongoing projects.** After an initial analysis, project areas for versions and changes can help you review how the documented understanding evolves as the codebase changes.

## Related Pages

- Create a New Project
- Navigate the Projects Hub
- Use Technical Docs

- Review Project Changes
- Compare Project Versions
- Export Documentation

## Troubleshooting

### You cannot find the project after creating it

Return to **Projects** and use **Search projects...**. If you have access to many projects, try **Search across all projects...**. Confirm that you used the intended project name during creation.

### The documentation does not answer a specific implementation question

Open **Technical Docs** to locate the relevant entity, then select **View Source** to inspect the supporting source material. Follow related entities, such as components, hooks, services, or repositories, rather than reviewing only the first item you find.

### You connected the wrong repository

From the project workspace, return to **Projects** using ← **Back to Projects**. Locate the incorrect project and use **Delete Project** if you have permission to remove it. Then select **New Project** and connect the correct repository before selecting **Create project**.