Search /

Tutorials

4 hacks to turbocharge your Cursor productivity

Published June 10th, 2025

by Tilde Thurium

Overview

Cursor, an AI code editor, is the fastest growing developer tool of all time, at least according to spearhead.co.

Cursor works pretty well out of the box, but there's a lot you can do to make it even better. In this post we'll cover four tips to make your Cursor agent more accurate, personalized, and better integrated with other developer tools.

If you haven't installed Cursor, follow the installation instructions here.

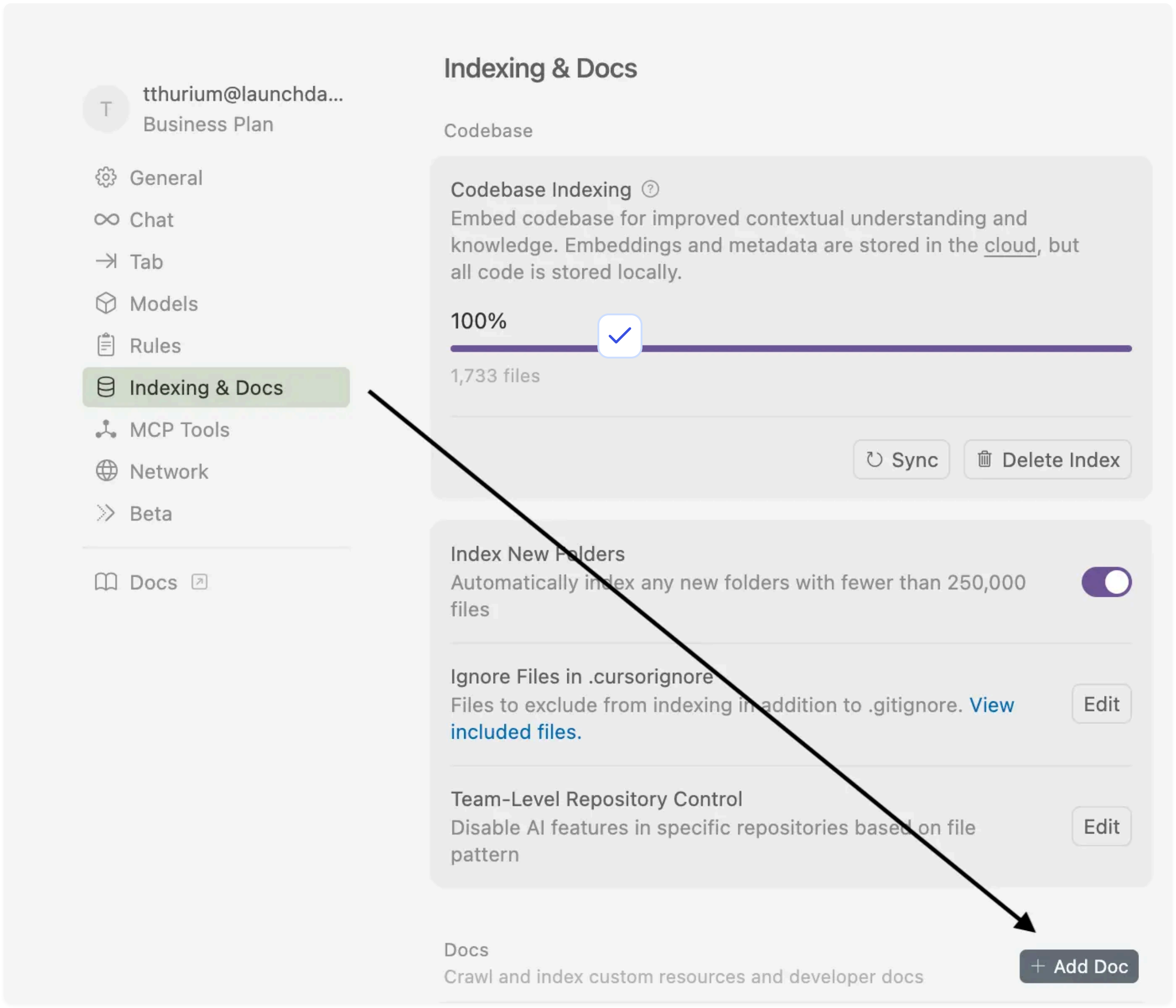
Tip #1: Add developer documentation as a data source

The more *context* an AI agent has, the better its suggestions will be. Generative model training data can be frozen months or years before the models are released to the public. This lag makes it hard for models to generate up-to-date code reflecting the latest changes to software libraries. Fortunately, you can add documentation to Cursor to enhance the coding agent's abilities and reduce hallucination.

If you add a root documentation link to Cursor, it will index the content of those docs, including child links. When you make queries to the AI agent, the AI agent will determine if

Cursor comes with documentation for many popular libraries already indexed. You can add documentation about specific libraries, tools, and SDKs you use in your codebase.

For example, I'll show you how to add the LaunchDarkly docs. In Cursor, go to **Settings** > **Cursor Settings** > **Indexing & Docs.** Click **Add Doc.**



Indexing & Docs

Codebase

Codebase Indexing
Embed codebase for improved contextual understanding and knowledge. Embeddings and metadata are stored in the cloud, but all code is stored locally.

100%

1,733 files

Index New Folders
Automatically index any new folders with fewer than 250,000 files

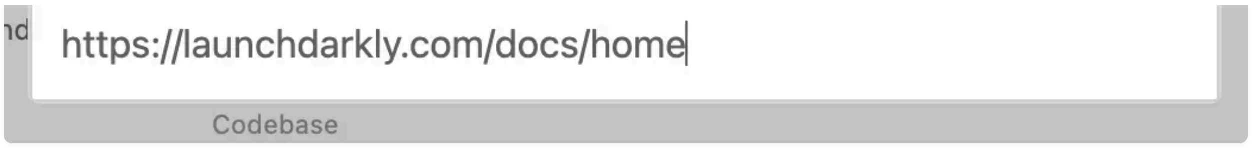
Ignore Files in .cursorignore
Files to exclude from indexing in addition to .gitignore. View included files.

Team-Level Repository Control
Disable AI features in specific repositories based on file pattern

Docs
Crawl and index custom resources and developer docs

The Add Doc button in the Indexing & Docs section of the Cursor settings page.

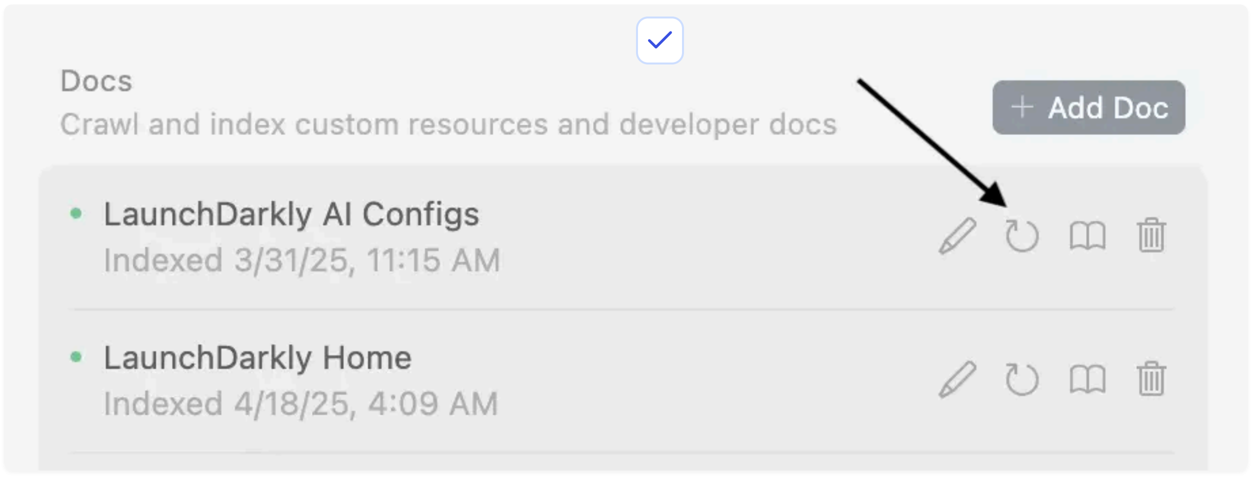
In the following dialog box, paste in the URL of the docs you'd like to add:



The add docs form.

Click *Add Docs*.

For better or for worse, software is never complete; thus, documentation is always in flux too. Cursor periodically reindexes any docs you have added. You can manually reindex by clicking the **curved arrow icon** in **Settings** > **Cursor Settings** > **Indexing & Docs.** You can also remove docs that are no longer relevant by clicking the **trash icon.**



Existing documentation that Cursor has indexed.

Tip #2: Use MCP to interact with APIs in plain English

Model-context protocol (MCP) is an open protocol that lets you interact with APIs using natural language.

For example, the LaunchDarkly MCP server lets you manage LaunchDarkly feature flags and AI Configs from within your IDE, in plain English.

LLM context windows are expanding, they aren't boundless. Adding and removing MCP servers doesn't take too long, so you can tinker with what servers are installed locally depending on the needs of your current engineering project. Developers are racing to build middleware layer that solves this problem, so it's likely this pain point is temporary.

How can you discover what MCP servers might be the most useful to you? MCP server registries are multiplying almost as fast as MCP servers. However, the most complete registry lives on the official *modelcontextprotocol* GitHub organization.

Tip #3: Use Cursor rules as system-level context

Cursor rules allow you to provide additional instruction to the coding agent. They are handy for specifying domain-specific knowled ✓

The rules are flexible enough that you can indicate when you want them to be attached to a query. The options are:

- **Always**: Be careful with this one, it can take up valuable context tokens for things that don't need to be included in every request.
- **Auto Attached**: You can instruct Cursor to auto-attach the rule when editing files matching a specific `*.extension`.
- **Agent Requested**: Describe the kinds of tasks the rule is helpful for, and the Cursor Agent will decide if the rule should be applied.
- **Manually**: You can manually add the rules to the context in your chat window with the AI agent.

Rules are written in Markdown. If you want to see an example of what they looks like in practice, this repository contains recommended Cursor rules for LaunchDarkly development.

You can share project-level rules with other developers on your team by committing them to a settings directory in a shared repository.

for personalizing your experience. Perhaps you find the AI agent's egregious overuse of exclamation points to be a bit much. You can add a Cursor rule to tone down the enthusiasm to a more manageable level.

```
When responding to queries, use a dry, matter-of-fact tone.
```

On the more unhinged end of the spectrum, you can try ramping up the absurdity:

```
Provide all code feedback in the form of a roast.
```

Tip #4: Pick the model that's best for your use case

Engineering is fundamentally about making tradeoffs. Model selection requires balancing quality, latency, and cost to suit your use case.

If you have an easier task to do, consider using a lighter-weight model. I find that Claude 3.5 Sonnet meets my everyday coding needs. Reasoning models with larger context windows such as OpenAI's o3 are more expensive, so consider saving those for gnarlier problems.

You can change your model in **Settings** > **Cursor Settings** > **Models**. Usage details currently live in **Settings** > **Cursor Settings** > **General Manage Account.**

New models are released daily. Keeping up to date is a struggle. For the latest info on model availability, see Cursor's documentation.

Conclusion

To get the most out of your Cursor experience:

- Add technical documentation as a data source for the Cursor agent.

- Install MCP servers locally to make interacting with your most-used APIs a breeze.
- Consider what size model best fits your use case.

Thanks for following along! If you have any questions, or you want to tell me about your favorite MCP servers, you can reach me via email (tthurium@launchdarkly.com), Bluesky, Discord, or LinkedIn.

Was this page helpful? Yes No

Previous

Create a feature flag in your IDE in 5 minutes with Launc... Next