Context

Rules

Control how the Agent model behaves with reusable, scoped instructions.

Rules allow you to provide system-level guidance to the Agent and Cmd-K AI. Think of them as a persistent way to encode context, preferences, or workflows for your projects or for yourself.

We support three types of rules:

Project Rules
Stored in `.cursor/rules` , version-controlled and scoped to your codebase.

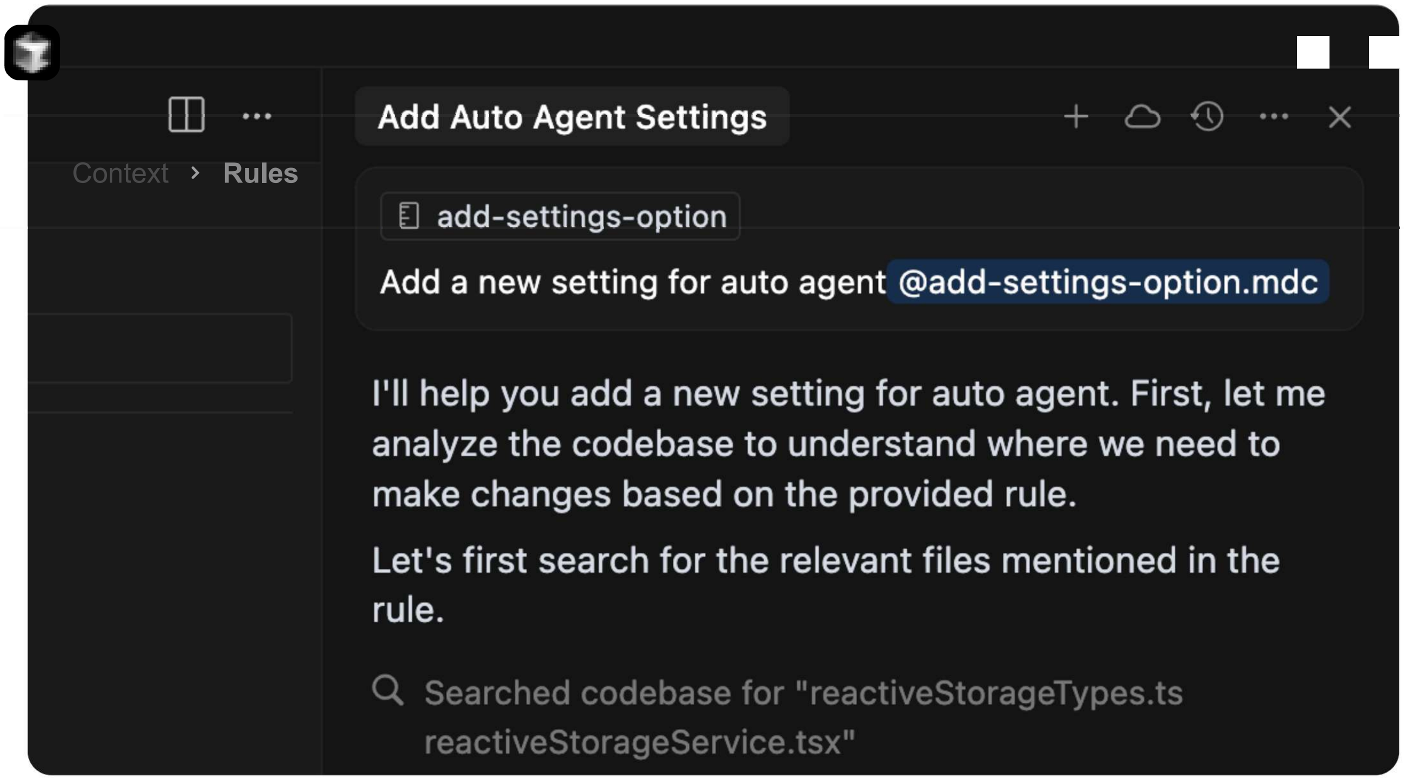
User Rules
Global to your Cursor environment. Defined in settings and always applied.

.cursorrules (Legacy)
Still supported, but deprecated. Use Project Rules instead.

How rules work

Large language models do not retain memory between completions. Rules solve this by providing persistent, reusable context at the prompt level.

When a rule is applied, its contents are included at the start of the model context. This gives the AI consistent guidance whether it is generating code, interpreting edits, or helping with a workflow.



Add a new setting for auto agent @add-settings-option.mdc

I'll help you add a new setting for auto agent. First, let me analyze the codebase to understand where we need to make changes based on the provided rule.

Let's first search for the relevant files mentioned in the rule.

Searched codebase for "reactiveStorageTypes.ts reactiveStorageService.tsx"

Rules apply to both **Chat** and **Cmd K**

Project rules

Project rules live in `.cursor/rules`. Each rule is stored as a file and version-controlled. They can be scoped using path patterns, invoked manually, or included based on relevance.

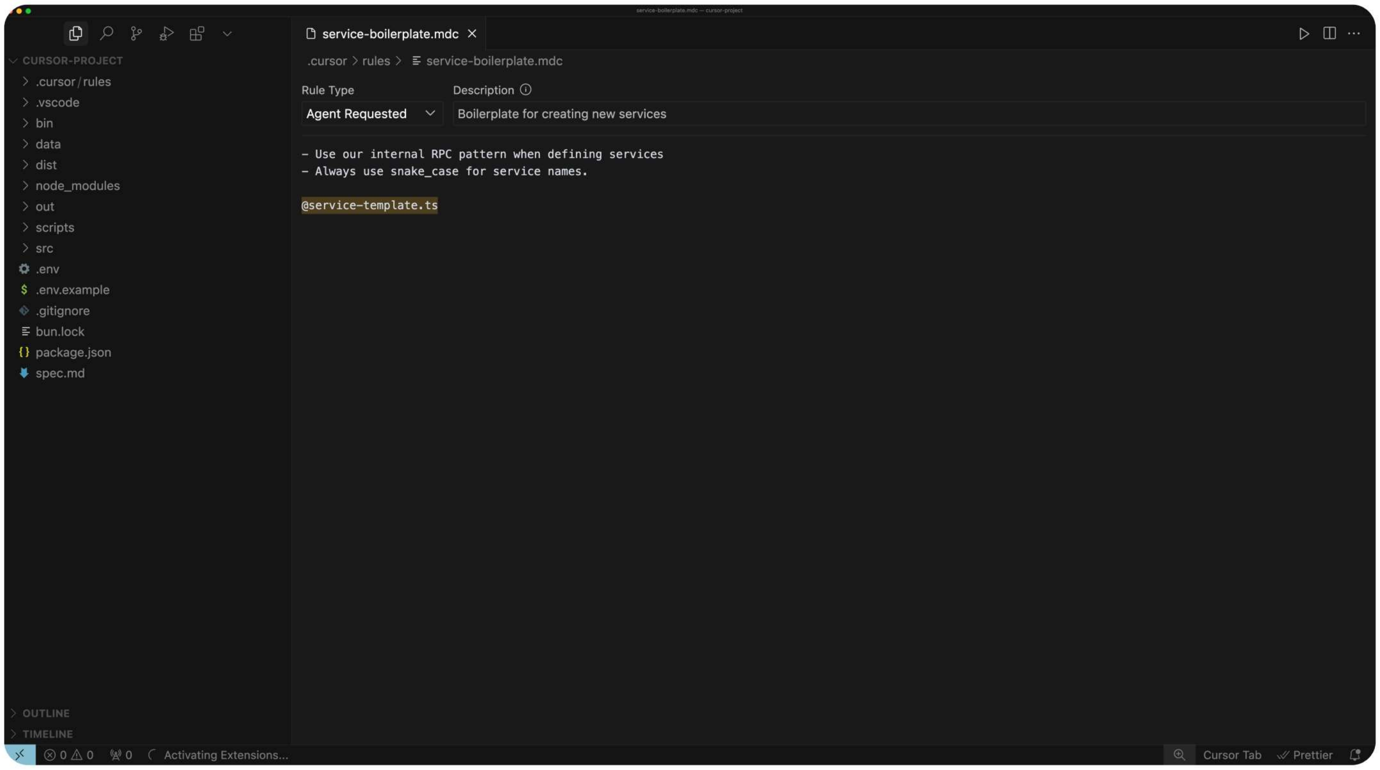
Use project rules to:

- Encode domain-specific knowledge about your codebase
- Automate project-specific workflows or templates
- Standardize style or architecture decisions

Rule structure

Each rule file is written in **MDC** (`.mdc`), a lightweight format that supports metadata and content in a single file. Rules supports the following types:

Rule Type	Description
`Always`	Always included in the model context
`Auto Attached`	Included when files matching a glob pattern are referenced
`Agent Requested`	Rule is available to the AI, which decides whether to include it. Must provide a description
`Manual`	Only included when explicitly mentioned using `@ruleName`



Rule Type: Agent Requested

Description: Boilerplate for creating new services

```
- Use our internal RPC pattern when defining services
- Always use snake_case for service names.

@service-template.ts
```

Example MDC rule

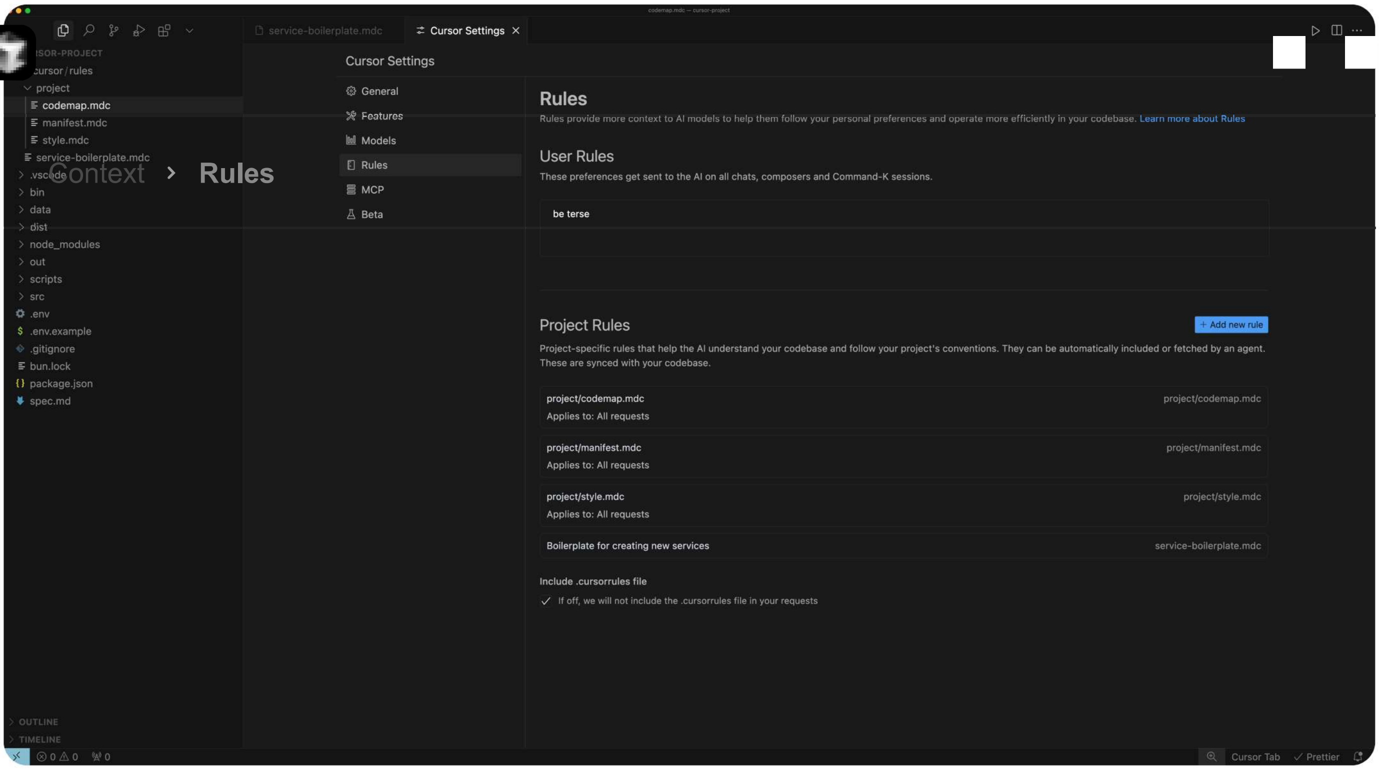
```
---
description: RPC Service boilerplate
globs:
alwaysApply: false
---

- Use our internal RPC pattern when defining services
- Always use snake_case for service names.

@service-template.ts
```

Referenced files like `@service-template.ts` will be included as additional context when the rule is triggered.

You can use `Cmd + Shift + P` > “New Cursor Rule” to create a rule quickly from inside Cursor.

Creating a rule

You can create a rule by using the `New Cursor Rule` command or going to `Cursor Settings > Rules`. This will create a new rule file in the `.cursor/rules` directory. From settings you can also see a list of all rules and their status.



Context > Rules

Cursor Settings

Rules

Rules provide more context to AI models to help them follow your personal preferences and operate more efficiently in your codebase. Learn more about Rules

User Rules

These preferences get sent to the AI on all chats, composers and Command-K sessions.

be terse

Project Rules

Project-specific rules that help the AI understand your codebase and follow your project's conventions. They can be automatically included or fetched by an agent. These are synced with your codebase.

project/codemap.mdc Applies to: All requests	project/codemap.mdc
project/manifest.mdc Applies to: All requests	project/manifest.mdc
project/style.mdc Applies to: All requests	project/style.mdc
Boilerplate for creating new services	service-boilerplate.mdc

Include .cursorrules file

✓ If off, we will not include the .cursorrules file in your requests

Best practices

Good rules are focused, actionable, and scoped.

- Keep rules concise. Under 500 lines is a good target
- Split large concepts into multiple, composable rules
- Provide concrete examples or referenced files when helpful
- Avoid vague guidance. Write rules the way you would write a clear internal doc
- Reuse rules when you find yourself repeating prompts in chat

Examples

Domain-specific guidance
Standards for frontend components and API validation

Boilerplate and templates
Templates for Express services and React components

Context › **Rules**

Workflow automation
Automating development workflows and documentation generation

From Cursor codebase

These are rules that we use internally at Cursor

Using Tailwind in Cursor

Adding a new setting in Cursor

There are many examples available from providers like Next.js, Cloudflare, and Browserbase. Community-contributed rules can be found across multiple crowdsourced collections and repositories online.

User rules

User rules are defined in **Cursor Settings > Rules**.

They apply to all projects and are always included in your model context.

Use them to:

- Set response language or tone
- Add personal style preferences

Example:

```
Please reply in a concise style. Avoid unnecessary repetition or filler language.
```

User rules do not support MDC, they are plain text only.

Team rules

There is no built-in way to share rules across projects today.

We plan to support shared, MDC-formatted rules that can be referenced across team projects. Until then, you can:

- Store shared rules in a dedicated repository
- Copy or symlink them into each project's `.cursor/rules` directory

`.cursorrules` (Legacy)

The `.cursorrules` file in the root of your project is still supported, but will be deprecated. We recommend migrating to the Project Rules format for more control, flexibility, and visibility.

FAQ

Why isn't my rule being applied?

Check the rule type. For `Agent Requested`, make sure a description is defined. For `Auto Attached`, ensure the file pattern matches the referenced files.

Can rules reference other rules or files?

Yes. You can use `@filename.ts` to include files in your rule's context.

Can I create a rule from chat?

Yes. Ask the AI to "turn this into a rule" or "make a reusable rule from this prompt".

Do rules impact Cursor Tab or other AI features? No. Rules are only given to the Agent and Cmd-K AI models.

Was this page helpful? Yes No

< Codebase Indexing Overview >

Product

Pricing

Downloads

Docs

Forum

Company

Careers

About

Security

Privacy

Resources

Terms

Changelog

Twitter

GitHub