ContextStream MCP Server: README

Contents

Overview 2
Your agent knows code. Give it the decisions behind yours. 2
Start free with MCP 2
Make the next session useful 2
Benchmarks 2
Data controls 3
Open source and contributing 4

Overview

ContextStream MCP Server

The context platform for AI agents.

Your agent knows code. Give it the decisions behind yours.

Connect your code, docs, and conversations so your agents can find the right files, recall saved decisions, and build on past work across sessions and tools.

The numbers, their sources, and what they do not claim are under Benchmarks.

Start free with MCP

10,000 monthly credits. No credit card required.

Works with Claude Code, Cursor, Codex, GitHub Copilot, Gemini CLI, Qwen Code, Kimi Code, Muse Code, ZCode, Zed, and more MCP clients, on Claude, GPT, Gemini, Kimi, GLM, Qwen, and other models. Create your account or sign in during MCP onboarding. No separate website signup needed.

macOS and Linux

```
curl -fsSL https://contextstream.io/scripts/mcp.sh | bash
```

Windows PowerShell

```
irm https://contextstream.io/scripts/mcp.ps1 | iex
```

Paste the command into your terminal and follow onboarding to connect your project and supported editor. Restart your editor after setup.

What gets installed and indexed? · Supported clients · Setup guide · Prefer to start on the web?

Make the next session useful

Ask your connected agent:

> Use ContextStream to find the files relevant to my next change. Cite the sources and retrieve any saved project decisions that should guide the work.

Then ask it to save a real decision and its reason. Start a new session and retrieve that decision without explaining it again. Indexing supplies code context; it cannot recover every undocumented decision.

Benchmarks

Each benchmark runs through the production API, is registered before its results are read, and is published with its method, confidence interval, and losses.

Benchmark	ContextStream	Compared with
LongMemEval-S, all 500 questions	**96.8%** (484/500); 96.0% on the half held out from tuning	Above Mastra (94.87%) and supermemory (85.4%) on the same official GPT-4o judge. Level with agentmemory's self-reported 96.2%, which is inside our 95% interval

Benchmark	ContextStream	Compared with
LoCoMo, all 1,540 questions	**95.7%** (1474/1540); 95.8% held out; 97.8% on the audited answer key	Above every system in the EverMemOS paper's same-judge table: EverMemOS 93.05%, Zep 85.22%, MemOS 80.76%, MemU 66.67%, Mem0 64.20%
Code search, 177 judged developer queries	**98.77%** best answer in the top 10; **100%** a useful file in the top 10	Embedding-only search 72.84% and ripgrep 42.59% on the same queries and judgments
Agent tasks, 8 tasks × 3 trials	**23/24** passed (96%); 9/9 on tasks that need team knowledge	14/24 (58%) and 0/9 with the repository alone

61 ms median server-side search time (127 ms p95) in the October 27 production run. End to end it was 223 ms, because the client ran in a different region from the one serving the workspace. Your latency depends on your distance from that region.

Competitor figures are as each vendor or the EverMemOS paper published them; we did not re-run them, and readers and judges differ across rows. The code-search baselines come from the August 1 run on the same queries. The agent-task result uses one repository and one agent model, so treat it as directional. Results, methodology, and limitations.

Data controls

Scoped access. Traceable sources. Configurable capture. Indexing sends eligible source contents to ContextStream for hosted search. Transcript saving and local Git metadata capture are on by default and can be turned off. Review data handling and controls before connecting sensitive projects.

Other installation options and troubleshooting

npm (Node.js 20+)

```
npx -y @contextstream/mcp-server@latest setup
```

The npm launcher downloads the native Rust binary for your platform and verifies its SHA-256 checksum. Pin an exact package version for production automation.

Hosted MCP without a local process

For clients supporting Streamable HTTP and OAuth, the hosted endpoint is:

```
https://mcp.contextstream.io/mcp?default_context_mode=fast
```

Supported clients lists one-line add commands for Claude Code, Codex, Gemini CLI, Qwen Code, Copilot CLI, and Droid, and one-click install for VS Code. Use the MCP documentation for other clients and stdio alternatives. A hosted connection alone does not sync your local checkout.

Preview or diagnose setup

```
contextstream-mcp setup --dry-run
contextstream-mcp doctor --scope=all --only-configured
```

Scripted and CI setup

For automation, wrap the install in `pipefail` so it fails when the download fails, instead of reporting success after installing nothing:

```
bash -o pipefail -c 'curl -fsSL https://contextstream.io/scripts/mcp.sh | bash'
```

`setup --yes` reads the key from `CONTEXTSTREAM_API_KEY` or saved credentials, and needs `--workspace-id` when the account has more than one workspace:

```
export CONTEXTSTREAM_API_KEY=...   # or: printf %s "$KEY" | contextstream-mcp configure
--api-key-stdin
contextstream-mcp setup --yes --editors=claude --workspace-id=<UUID> --project-path=.
```

Package Bouncer (opt-in)

A hook for Claude Code and Cursor that checks the packages an install command names against their public registry before the command runs. It covers npm, pnpm, yarn, bun, npx, pip, uv, uvx, pipx, poetry and cargo, including `claude mcp add ... -- npx ...`.

- It **denies** a package the registry does not have, naming the popular package it looks like. An agent may have invented the name, and anyone can register it.
- It **asks** before a barely downloaded lookalike of a popular package.
- It **allows** everything else, including whenever a lookup fails or takes too long.

It skips projects and commands that use a private registry (`.npmrc`, `pip.conf`, uv or poetry sources, `--registry`, `--index-url`). `CONTEXTSTREAM_PACKAGE_BOUNCER=off` turns it off.

Claude Code: add this entry to `PreToolUse` in `~/.claude/settings.json`, next to the hooks setup installed. `update-hooks` leaves it alone.

```
{ "matcher": "Bash", "hooks": [{ "type": "command", "command": "contextstream-mcp hook
package-bouncer", "timeout": 8 }] }
```

Cursor (`~/.cursor/hooks.json`):

```
{ "version": 1, "hooks": { "beforeShellExecution": [{ "command": "contextstream-mcp hook
package-bouncer" }] } }
```

Open source and contributing

This repository contains the MIT-licensed Rust MCP server, client, editor setup, and release tooling. The hosted backend is separate and is not included here. See the architecture, release integrity, and latest release.

```
cargo build --locked -p mcp-server --bin contextstream-mcp
cargo test --locked --workspace
```

Use the pinned Rust toolchain. For contribution checks and commit sign-off, read CONTRIBUTING.md. Report vulnerabilities using SECURITY.md. See LICENSE, NOTICE, and GOVERNANCE.md for licensing, trademarks, and project governance.

Let humans be human.

Optional deep project learning

Deep project learning starts off for new accounts. Plain `contextstream-mcp setup --yes` preserves your current choice. To review consent, run `contextstream-mcp configure --account-learning on` and confirm on the signed-in privacy page. To withdraw, run `contextstream-mcp configure --account-learning off`. Agents cannot consent for you. Doctor and init show status only. See data handling and controls.