MCP SERVER | NO CODE | CLOUD HOSTED

SonarQube & SonarCloud MCP

Check Code Quality and Security in Chat.

SonarQube & SonarCloud MCP brings professional code quality analysis directly into your AI agent's workflow. Stop hunting through browser tabs to find vulnerabilities, technical debt reports, or test coverage metrics. This MCP lets you diagnose complex codebase issues—from security hotspots to duplication ratios—using plain language queries against self-hosted or cloud static analysis results.

A+ Quality Score 100/100

static-analysis | code-quality | bug-detection | technical-debt | on-premise | code-security

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/sonarqube-sonarcloud

SCAN TO VIEW

The connectivity layer between AI and the world's software.

Vinkius sits between AI and every application. All communication passes through Vinkius Cloud via the Model Context Protocol (MCP) — with governance, observability, and security at every layer.

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the connectivity layer where AI connects to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

Four Pillars of the Vinkius Runtime

01 — Security by design

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

02 — Built on MCP Fusion

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

03 — Deterministic observability

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

04 — Autonomous operations

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

AES-256	Ed25519	24h TTL	V8 Isolate
Encryption at rest	PKI vault signatures	Ephemeral session keys	Sandboxed execution

One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

01 — Ed25519 PKI Vault

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

02 — V8 Isolate Sandboxing

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

03 — SSRF Guard

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

04 — DLP & PII Redaction

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

05 — Cryptographic Audit Trail

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

06 — Honeytoken Trap System

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

01 — Server deactivated

The MCP server is immediately taken offline across the entire cluster.

02 — All tokens revoked

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

03 — WebSocket connections killed

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

Control Plane

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

Agent Activity

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

FinOps

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

Tool Health

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

Firewall & DLP

PII redaction activity, sensitive data protection counters, and security event timeline.

Incident Log

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

SonarQube & SonarCloud MCP

10 tools available
Cloud-hosted on Vinkius

Diagnosing code flaws used to mean juggling multiple dashboards and context switches every time you needed a single metric. Now, you can connect your self-hosted SonarQube instance or SonarCloud dashboard right into your AI client through Vinkius. Your agent talks directly to the analysis engine. Instead of manually filtering logs or running complex CLI commands, you simply ask for details—like finding all Critical security issues across a project or checking if the Quality Gate passed. You can pull raw code lines from specific components, measure test coverage, and even audit which rules were enabled without ever leaving your chat window. It turns massive technical debt reports into simple conversational facts.

Core Capabilities

01 — Check Code Health Status

Your agent verifies the overall quality gate status or retrieves specific code metrics, like unit test coverage and complexity indexes.

02 — Hunt Security Flaws

You pinpoint exact security vulnerabilities by filtering issues based on severity (Critical, Blocker, Major) or finding manually marked security hotspots in the codebase.

03 — Map Code Structure and Debt

The system provides a hierarchical view of all files and directories in the project and calculates code duplication levels for specific components.

04 — Review Source Code Details

You retrieve raw, annotated source code lines or list all active analysis rules to understand exactly what was checked during the build process.

05 — Discover Projects and Components

The agent helps you find project keys and map out the entire component tree structure of your application.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/sonarqube-sonarcloud** — connect your AI agent in three steps.

01 Subscribe to this MCP in Vinkius, providing the necessary connection URL for your self-hosted or cloud SonarQube instance.

02 Securely inject your required API Token into your AI client's configuration and authorize the connection.

03 Use plain language prompts with your AI agent—for example, 'What is the quality gate status of project X?'—to execute deep analysis queries.

The bottom line is that you get instant access to data points previously locked behind multiple web dashboards and command-line interfaces.

Built For

This MCP is for any engineering role constantly battling technical debt or needing immediate, actionable security feedback. It's perfect for the developer who hates switching between GitHub, Jira, and SonarQube to approve a simple merge request.

Software Engineer

You use this MCP to ask your agent why a Pull Request failed its quality gate check and demand the exact code changes needed for approval.

DevSecOps Specialist

You query specific details on critical CVEs or search issues by severity before approving any production merge, ensuring compliance is met automatically.

Tech Lead

You gather project duplication ratios across multiple modules or map the entire component structure to audit overall system health and technical debt.

What Changes When You Connect

01 Stop wasting time context switching. You can ask your AI agent for the `get_quality_gate_status` directly, getting an immediate pass/fail report without opening a single browser tab.

02 Pinpoint security risks instantly. Use `search_issues` to filter code flaws by severity level (Critical, Blocker) and immediately know where to focus your refactoring effort.

03 Measure technical debt with precision. Running the `get_measures` tool gives you actionable numbers on test coverage and tech debt rates across core services.

04 Understand the entire codebase structure using `get_component_tree`. This lets you audit project dependencies and map out every file before starting development.

05 Deep dive into code flaws by running `get_hotspots`. You find exactly which lines of code need a human eye, saving time on false positives.

Real-World Applications

Investigating PR Failures

A developer knows their merge failed because the Quality Gate didn't pass. They prompt their agent: 'What are the top three issues preventing merging on Project X?' The agent runs `search_issues`, finds a Critical issue, and pulls the relevant component details via `get_component_tree`.

Pre-Audit of Legacy Code

A tech lead is assigned to an old service. They prompt: 'Show me all code duplication in the user authentication module.' The agent uses `get_duplications` and presents a report, instantly quantifying the technical debt before any work begins.

Security Vulnerability Deep Dive

A DevSecOps engineer needs to confirm if a specific payment processing file has known security flaws. They ask the agent to run `get_hotspots` against the component, getting line numbers and rule IDs for immediate investigation.

Reporting Technical Debt

A team lead needs to report on overall code quality during a quarterly review. They prompt: 'What is the current branch coverage and tech debt rate?' The agent runs `get_measures` and provides clear, quantifiable metrics.

Patterns to Avoid

Searching for data manually

✗ AVOID

The engineer opens SonarQube in the browser, navigates to 'Security Issues,' applies filters for 'Critical' severity, and then copies the details into a document.

✓ INSTEAD

Instead, prompt your agent: 'Show me all Critical issues for Project X.' The agent runs `search_issues` and delivers the filtered list directly in the chat.

Relying on general knowledge

✗ AVOID

A developer thinks they know where a vulnerability exists but can't pinpoint the file or line number without guessing.

✓ INSTEAD

Use `get_hotspots` to force the agent to identify the exact component and line source area, providing concrete coordinates for the fix.

Ignoring structural context

✗ AVOID

A developer is looking at a file but doesn't know which module it belongs to or if other parts of the system use similar code.

✓ INSTEAD

Run `get_component_tree` first. This maps out all files and directories, giving you the necessary structural context before diving into specific code details.

The Right Fit

Use this MCP if your workflow requires integrating deep, structured static analysis results—like test coverage metrics or security vulnerability lists—into natural conversation. You need to move beyond simple status checks; you need quantitative data points like duplication ratios (`get_duplications`) and specific code snippets (`get_source_code`). Don't use this MCP if all you need is a basic list of projects; for that, just run `search_projects` . Also, don't use it if you are only interested in high-level CI/CD status checks, as the agent requires more than just a simple gate check to deliver maximum value.

The Friction of Code Quality Audits Today

When you need to understand why a PR failed or if a service is technically sound, you currently have to jump through hoops. You check the SonarQube dashboard for an overall grade, then click into 'Issues' and filter by severity. If that doesn't help, you might manually drill down into component hierarchies just to find the file name, only to copy it over to another tool for context.

With this MCP, all that friction disappears. You simply ask your agent: 'What are the Critical flaws in the user module?' It runs `search_issues` and delivers a structured list with exact details—the component name, rule ID, and line number—all without you leaving the chat.

Getting Deep Insight with SonarQube & SonarCloud MCP

You no longer have to manually run separate checks for coverage versus debt. You can ask your AI agent to gather `get_measures` across the whole project, instantly comparing branch coverage against the technical debt rate in one conversational response.

It's not just about getting data; it's about making that data actionable. Your agent connects the dots —it finds a security hotspot using `get_hotspots` , and then you can immediately use `get_source_code` to review the exact lines needed for remediation.

SonarQube & SonarCloud: 10 Tools

These ten tools let you query every aspect of a codebase—from project structure to specific lines of duplicated code—all through your AI client.

#	TOOL	DESCRIPTION
01	`get_component_tree`	Gets a full, structured list of all files and directories within the SonarQube project, along with key metrics for each component.
02	`get_duplications`	Calculates the amount of repeated code blocks found in a specified file across the entire codebase.
03	`get_hotspots`	Identifies and lists specific sections of code that carry elevated security risk or require manual human review.
04	`get_measures`	Retrieves key metrics, such as test coverage percentages (branch/line) and technical debt index, for the entire project.
05	`get_quality_gate_status`	Checks if a project has passed or failed its defined quality standards, returning a simple pass/fail status.
06	`get_source_code`	Retrieves annotated source code lines from SonarQube for a specified file path.
07	`list_quality_gates`	List all quality gate definitions in SonarQube
08	`list_rules`	Provides a list of all analysis rules that are enabled and active on your organization's platform.
09	`search_issues`	Searches for code issues across the project, allowing you to narrow results by severity (like Critical or Major).
10	`search_projects`	Finds project keys and names across your entire organization's SonarQube or SonarCloud setup, which are required for most other functions.

See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

U Search our primary repository and give me the official Quality Gate diagnostic.

Running checks on `backend-main-api` using `get_quality_gate_status`.
Gate Result: OK 🟢

The most recent static checks successfully validated. Thresholds for Reliability facing bugs and Security ratings are comfortably within standard tolerance boundaries.

U Run a test coverage and technical debt measure retrieval on all core services.

Invoked `get_measures` spanning metric keys `coverage,sqale_index` across your `core-srv-module` baseline.

- **Branch Coverage**: 85.3%
- **Line Coverage**: 81.6%
- **Tech Debt Rate**: 14h 22min (Sqale Index)

Do you want me to search issues to drop technical debt in half?

U Tell me the precise component lines hitting security hotspot alerts.

Executing `get_hotspots` against origin `frontend-app-portal`:

1. 🔴 **Vulnerability Risk Level 1** → Located in `pages/api/submit.js` (Component File)
- **Rule ID**: S1452 (Unvalidated Redirect / SSRF potential)
- **Line Source Area**: ~ L: 47

2. 🔴 **Vulnerability Risk Level 1** → Located in `utils/hasher.js`
- **Rule ID**: S2278 (Weak cryptographic algorithm used for generic salts)

I can retrieve lines specifically natively via `get_source_code` if you're ready to fix this token loop now.

Frequently Asked Questions

01 **How do I find out what projects are available in SonarQube with the SonarQube & SonarCloud MCP?**

You use the `search_projects` tool. This function scans your entire organization's setup and returns a list of project keys and names, which you then need to pass to other tools for analysis.

02 **Can I check if my code passed quality standards using SonarQube & SonarCloud MCP?**

Yes, run the `get_quality_gate_status` tool. It gives an immediate status update (Pass/Fail) on whether your current build meets all defined quality requirements.

03 **How does the SonarQube & SonarCloud MCP help with code duplication?**

You use the `get_duplications` tool. This analyzes a specific file and quantifies exactly how many blocks of code are duplicated across your project, helping you target refactoring efforts.

04 **What is the best way to find vulnerabilities using this MCP?**

Start by running `search_issues`, filtering results by Critical or Blocker severity. If you need more detail on a specific risk, use `get_hotspots`.

05 **Does the SonarQube & SonarCloud MCP require me to know API details?**

No. You only need plain English prompts directed at your agent. The agent handles calling the specific tools, like `get_measures`, using the required project keys in the background.

Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

CLIENT	WHERE TO CONFIGURE
Claude AI	Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint
Cursor	Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint
VS Code	Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"sonarqube-sonarcloud": { "url": "…" }`
Windsurf	MCP Settings → `mcp_settings.json` → Add endpoint URL
ChatGPT	Settings → Tools & plugins → Add MCP server → Paste endpoint
Gemini	Extensions → Add MCP Server → Paste endpoint URL

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

SonarQube & SonarCloud is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

Start at cloud.vinkius.com →

vinkius.com · support@vinkius.com

INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by SonarQube & SonarCloud. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

DOCUMENT INFORMATION

Generated	June 2026
MCP Server	SonarQube & SonarCloud MCP
Server ID	`019d760b-1b55-7386-8aa7-f737c45b64df`
Platform	Vinkius Cloud for AI Agents
Endpoint	`https://edge.vinkius.com/{token}/mcp`

LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/mcp/sonarqube-sonarcloud.