MCP SERVER | NO CODE | CLOUD HOSTED

BugBug MCP

Automate QA and Regression Testing by Conversation

BugBug connects automated browser testing and quality assurance directly into your workflow. List and run high-fidelity end-to-end tests in the cloud using natural conversation, finding regressions before users do.

A+ Quality Score 98.33/100

browser-automation | end-to-end-testing | no-code-testing | web-testing | quality-assurance | regression-testing

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/bugbug

SCAN TO VIEW

The infrastructure that powers AI agents in the real world.

Vinkius connects AI to the world's software through secure, enterprise-grade infrastructure — enabling real-world execution at scale, built on the Model Context Protocol (MCP).

Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the cloud infrastructure where AI agents connect 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.

BugBug MCP

12 tools available
Cloud-hosted on Vinkius

This MCP lets you manage complex web application quality checks without touching a dashboard. You can tell your agent to execute full test suites across different environments or simply check if a single login flow still works after a deployment. It handles everything from listing all available projects and running specific tests, to pulling the detailed reports afterward. Instead of digging through console logs for hours, you ask your AI client, and it gets the status update instantly. Since Vinkius hosts this MCP, you connect once to your agent—whether that's Claude or Cursor—and gain full control over monitoring platform-wide quality in real time. Your AI acts like a dedicated QA engineer, handling test orchestration and historical performance tracking using only natural conversation.

Core Capabilities

01 — Execute Test Suites

Trigger an entire group of automated tests to run simultaneously and monitor the overall quality results.

02 — Run Specific Tests

Initiate a single, isolated test case for quick verification after making small code changes.

03 — Retrieve Run Statuses

Check the real-time status and detailed reports of any completed or running test job.

04 — Manage Test Inventory

List all available projects, suites, and individual tests to see what needs checking.

05 — Secure Network Setup

Fetch the specific IP addresses used by BugBug for secure firewall allowlisting.

One Click on Vinkius — From Prompt to Execution

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

01 Subscribe to this MCP on Vinkius and retrieve your API Token from your BugBug dashboard's Integrations tab.

02 Connect your preferred AI client (like Cursor or Claude) using the provided token.

03 Ask your agent a direct question, such as 'Run the checkout suite in staging,' letting it handle the complex execution logic.

The bottom line is you talk to your agent like you're talking to a teammate; it handles all the API calls and reporting for you.

Built For

This MCP is essential for QA Engineers who are tired of clicking through complex dashboards just to verify a single feature. It helps developers validate code quality immediately after pushing a build, and DevOps teams that need reliable network information.

QA Engineer

Triggers full regression suites via natural language commands and pulls historical data on test failures to coordinate QA trends.

Software Developer

Verifies application quality immediately after a local deployment without leaving their IDE or writing boilerplate API calls.

DevOps Engineer

Automates the retrieval of testing IPs and monitors large suite performance to ensure network access rules are up-to-date.

What Changes When You Connect

01 Speed up debugging. Instead of manually checking logs, you can ask the agent to run a specific test or suite using `run_test` or `run_suite`, getting immediate status reports.

02 Simplify monitoring. You don't need to navigate multiple dashboards; simply use `list_suite_runs` and `get_suite_run` to get an overview of platform-wide quality trends instantly.

03 Keep your network secure. Use the `get_ips` tool to programmatically retrieve all necessary BugBug IP addresses for firewall allowlisting, eliminating manual lookups.

04 Deep dive into failures. If a test fails, you can use `get_test_run` and `get_test` to pull detailed execution statuses and high-fidelity reports without leaving your chat interface.

05 Maintain history effortlessly. You can check historical records using `list_test_runs`, giving you the data needed for quarterly quality trend analysis.

Real-World Applications

Need to verify a critical path after a minor update

A developer pushes a small backend fix. Instead of running 30 separate manual checks, they prompt their agent: 'Run the checkout process suite.' The agent executes the `run_suite` tool and reports back if the entire flow passed.

Debugging an intermittent network block

A DevOps engineer needs to ensure the new staging environment is accessible. They ask their agent to run `get_ips`, instantly providing the required IP addresses for the firewall team, saving hours of manual credential exchange.

Checking performance regressions on a specific feature

A QA engineer notices slow load times in one area. They prompt their agent: 'Check the status and results for test ID 456.' The agent uses `get_test_run` to retrieve detailed reports, pinpointing exactly which step failed.

Reviewing overall platform stability

A team lead wants a quick view of recent quality efforts. They ask the agent to list all recent suite runs using `list_suite_runs`, getting an immediate, aggregated summary without visiting the dashboard.

Patterns to Avoid

Trying to manually gather IP addresses

✗ AVOID

The DevOps team spends 30 minutes emailing a request to BugBug support just to get a list of IPs for firewall rules, causing deployment delays.

✓ INSTEAD

Use the `get_ips` tool. Your AI client fetches all necessary BugBug IP addresses directly through conversation, making it instant and auditable.

Manually triggering test runs

✗ AVOID

The QA engineer has to log into the web portal, find the suite, select an environment, and hit 'Run'—a multi-step process prone to human error.

✓ INSTEAD

Use the `run_suite` tool. Your AI client handles the entire execution trigger with a single natural language command.

Guessing which test ran last

✗ AVOID

The developer can't remember if they checked the 'Login Flow' or the 'Search Bar' yesterday and has to manually sift through pages of logs.

✓ INSTEAD

Use `list_test_runs` and then specify the exact tool name in your query. The agent retrieves the specific test run ID you need.

The Right Fit

Use this MCP if your primary bottleneck is managing, monitoring, or triggering automated web testing cycles for multiple environments. If you constantly find yourself clicking through dashboards to check status logs, retrieve IP ranges, or trigger full regression suites, this tool solves that workflow friction.

However, don't use it just because your agent can send emails or manage calendars. This MCP is strictly about application quality and testing infrastructure; its tools (`run_suite` , `get_ips` , etc.) do nothing else. If you only need to read a simple data record (like reading a user profile), other generic database connectors are better suited.

Web QA requires too many clicks.

Today, verifying application quality is a process of manual labor. You jump between the BugBug dashboard, copy run IDs into spreadsheets, and click through multiple tabs to check if a suite passed in staging or production. Getting IP addresses for firewall rules? That means writing a ticket and waiting hours for an email reply.

With this MCP, you simply talk to your agent. Need to know the status of the 'User Profile' test run? Ask it. Get the IPs needed for deployment? Ask it. Your AI client handles all that complexity behind the scenes, giving you immediate answers without a single manual click.

BugBug MCP: Run tests with natural language.

Forget having to find the exact test suite name or remember which run ID was most recent. You don't need to navigate `list_suites` and then manually select an environment from a dropdown menu; you just tell your agent what you want done, and it figures out the correct parameters.

The whole process is now conversational. Your time moves from dashboard navigation to actual code writing. It’s that simple.

BugBug: 12 Tools for Automated Testing

These tools let you list projects, trigger specific test runs, retrieve IPs, and monitor the status of your automated web application tests.

#	TOOL	DESCRIPTION
01	`get_ips`	Retrieves the specific BugBug IP addresses for firewall allowlisting purposes.
02	`get_suite_run`	Checks the status and details of an entire test suite execution run.
03	`get_suite`	Retrieves detailed information about a specific, defined test suite.
04	`get_test_run`	Gets the status and results for an individual test run execution.
05	`get_test`	Retrieves detailed information about a specific, defined automated test case.
06	`list_projects`	Lists all existing projects configured within your BugBug account.
07	`list_suite_runs`	Fetches a list of the most recent completed test suite runs for review.
08	`list_suites`	Lists all defined and available automated test suites.
09	`list_test_runs`	Fetches a list of the most recent completed individual test runs for review.
10	`list_tests`	Lists all defined and available automated tests within your projects.
11	`run_suite`	Triggers an immediate, live run of a specified test suite in the cloud.
12	`run_test`	Initiates a single, focused execution for a specific automated test case.

See It in Action

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

U List all my automated tests in BugBug.

I've retrieved your test directory. You currently have 15 automated tests, including 'Login Flow' (ID: test_123) and 'Checkout Process'. Which one would you like to execute or inspect?

U Run the 'Smoke Test' suite (ID: 'suite_456') in the 'Staging' environment.

Suite execution triggered! I've initiated the 'Smoke Test' suite in your staging environment. The run ID is 'run_789'. I'll monitor the progress and notify you as soon as the results are ready.

U Show the status and results for the latest test run ID 'run_789'.

Fetching run results... Execution run_789 has successfully 'Passed'. All 5 steps were verified. You can view the full high-fidelity report here: [report_url]. Shall I check the historical trends for this test?

Frequently Asked Questions

01 **How do I use BugBug MCP to check my network IPs?**

You call the `get_ips` tool by asking your agent for the current IP addresses. The agent retrieves all necessary BugBug IP ranges directly and presents them to you, ready for copy-pasting into firewall rules.

02 **Can I use BugBug MCP to run a whole test suite?**

Yes, simply tell your agent which suite you want executed. It uses the `run_suite` tool to trigger the job in the cloud and gives you an immediate confirmation ID.

03 **What if I only need to check one specific test?**

You can use the `run_test` tool by asking your agent. You specify the name or ID of the single test case, and it initiates a quick run without activating an entire suite.

04 **How do I check the status of a past test?**

To see historical results, you can ask your agent to use `list_test_runs` first. Then, provide the ID to get the detailed status and full report for that run.

05 **Does BugBug MCP help me manage my projects?**

Yes, it handles project visibility using the `list_projects` tool. You can ask your agent to list all available projects so you know what testing areas are configured in your account.

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 `"bugbug": { "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

BugBug 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 BugBug. 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	BugBug MCP
Server ID	`019dd0c7-26c8-73cb-b7c9-27065ca5666a`
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/bugbug.