MCP SERVER | NO CODE | CLOUD HOSTED

Supabase MCP

Manage projects, databases & secrets by conversation

Supabase manages your entire backend infrastructure through conversation. Use this MCP to list, create, and manage projects across different regions, configure Postgres settings like max connections, and handle all project secrets—all without leaving your AI client.

A+ Quality Score 100/100

postgres | backend-as-a-service | database-management | secret-management | branching | cloud-deployment

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

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.

Supabase MCP

14 tools available
Cloud-hosted on Vinkius

Managing a complex backend stack usually means jumping between dashboards: checking project status in one tab, updating environment variables in another, and adjusting database limits on a third. This MCP lets you take full control of your entire Supabase setup using nothing but natural conversation.

Your agent acts like a dedicated DevOps engineer right inside your IDE. You can list all organizations or check the details for a specific project reference. Need to prepare an environment? You'll create development branches with isolated databases instantly. You also get deep control over Postgres settings, allowing you to view and update parameters like work_mem. If you need better security, you can manage API keys and credentials by creating or deleting secrets safely. It's the complete backend toolkit that lets you focus on coding instead of clicking through administrative dashboards. We built this MCP so your AI client gets access to everything, all organized within the Vinkius catalog.

Core Capabilities

01 — Project and Organization Discovery

List all organizations or retrieve detailed information about any specific Supabase project.

02 — Backend Lifecycle Management

Pause unused projects to save money, restore data from a paused state, or create new projects in different regions.

03 — Database Environment Isolation

Create isolated development branches for testing features without impacting the main production database.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to this MCP and provide your Supabase Personal Access Token.

02 Next, tell your AI agent what you need done—for example, 'List all my projects in the EU region' or 'Update max_connections.'

03 Finally, the MCP executes the action via the API and returns a clear confirmation of the changes made to your backend.

The bottom line is that you talk to your AI client, and it handles all the complex API calls needed to manage your entire backend stack.

Built For

This MCP targets developers who get frustrated by context switching. It's for the DevOps engineer tired of logging into five different dashboards just to check project status or adjust a database setting, and the team lead needing an immediate audit view across multiple organizational projects.

Backend Developer

Uses this MCP to provision new development branches for feature testing and manage environment secrets without leaving their IDE.

DevOps Engineer

Audits organizational health by listing all projects, pausing unused resources with `pause_project` to save costs, and checking Postgres configuration details.

Solutions Architect

Manages the overall lifecycle of multiple client backends, provisioning new projects or restoring data for clients using `create_project`.

What Changes When You Connect

01 You can provision a new development branch using `create_branch` and get an isolated copy of the database instantly. This means your team can test big features without fear of breaking production data.

02 Stop paying for idle resources. Use `pause_project` to suspend entire projects when they aren't in use, saving money while ensuring all your core data remains perfectly intact.

03 Maintain total security control by using `create_secret` and `list_secrets`. You never have to manually update or copy sensitive credentials across different files or platforms again.

04 Tweak the database performance directly. With `get_postgres_config`, you can see current settings, and then use `update_postgres_config` to change parameters like max connections without writing complex SQL scripts.

05 The whole process is centralized. Instead of clicking through multiple dashboards, your AI agent handles project discovery with `list_organizations` and getting deep details with `get_project`, all in one conversation flow.

Real-World Applications

Scaling up a new feature requires a fresh environment.

A developer needs to test an OAuth change. They ask their agent to 'Create a dev branch called auth-test for the staging API.' The agent executes `create_branch`, giving them a safe, isolated copy of the database immediately.

The quarterly audit shows unused projects costing money.

A DevOps engineer asks their agent to 'Find all inactive projects and pause them.' The agent runs `list_projects` and then calls `pause_project` for every dormant entry, stopping costs automatically.

Need to adjust database performance limits.

A team lead notices slow query times. They ask their agent to 'Check the current Postgres config and increase work_mem.' The agent uses `get_postgres_config` and then executes `update_postgres_config`, adjusting the settings in one go.

Migrating a project requires new credentials.

A solutions architect needs to secure an API endpoint for a client. They ask their agent to 'Create a new secret called billing-api key.' The agent uses `create_secret` and confirms the value is encrypted at rest.

Patterns to Avoid

Assuming general database connectivity

✗ AVOID

Trying to use generic cloud tools to change connection pooling settings, only to realize they lack specific Supabase parameters.

✓ INSTEAD

Always use this MCP's specialized tools. For instance, instead of a generic update command, use `update_postgres_config` and pass the exact JSON object for 'max_connections'.

Manually tracking project status

✗ AVOID

Spending time clicking through individual project dashboards just to see which ones are active or paused.

✓ INSTEAD

Simply ask your agent to `list_projects` and include criteria like 'show me all projects that are paused' to get a clean, actionable list.

Handling secrets in plain text

✗ AVOID

Copying API keys into documentation or sharing them via chat instead of using a controlled vault.

✓ INSTEAD

Manage credentials using `create_secret` and `list_secrets`. The system handles encryption, meaning you never expose the value outside of the secure MCP boundary.

The Right Fit

Use this MCP if your primary pain point is managing backend infrastructure complexity across multiple services—specifically project lifecycles, database parameters, or security secrets. It's built for operational tasks like pausing projects (`pause_project`), creating isolated dev environments (`create_branch`), and tuning Postgres settings using `update_postgres_config` . Don't use this if

you simply need to read raw data from a single table; for that, your agent should query the database directly. This MCP is an operations layer, not a pure data retrieval tool. If all you need is simple listing (like `list_users`), other general-purpose API connectors might suffice, but when you need full lifecycle control and deep configuration access, this MCP is necessary.

Backend administration used to be a painful clickfest.

Today, checking on your backend requires context switching. You open the main Supabase dashboard to see if Project A is running, then jump into the settings tab for Postgres config just to check `work_mem`, and finally, you have to navigate to a separate 'Secrets' area to verify credentials. It’s tedious, slow, and easy to miss something critical.

With this MCP, you tell your agent what you need done—for example, 'I need to audit all projects in the EU region.' You get an immediate list of everything necessary, eliminating manual clicks and cutting your setup time from twenty minutes down to seconds.

Gain full control over project structure with Supabase Alternative MCP

Before this, setting up a dev environment meant creating a whole new database instance and manually linking it. Now, you ask your agent to `create_branch`, and the system handles provisioning that isolated copy instantly.

You're no longer limited by the dashboard's UI flow. You talk directly to the backend API through natural language. It’s fast, reliable, and keeps your entire development process inside your AI client.

Supabase Alternatives: 14 Tools for Backend Ops

These tools allow you to manage the entire lifecycle of a Supabase project—from provisioning new databases to updating deep configuration settings via natural conversation.

#	TOOL	DESCRIPTION
01	`create_branch`	Creates an isolated copy of your database for development work on a specific project.
02	`create_project`	Provisions an entirely new Supabase project, including Postgres, Auth, and Storage services.
03	`create_secret`	Securely adds a new API key or credential to the project's environment secrets.
04	`delete_secret`	Permanently removes an existing secret from the project; this action cannot be undone.
05	`get_user`	Checks which Supabase user is currently authenticated to confirm token access.
06	`get_postgres_config`	Retrieves the current operational settings for the project's PostgreSQL database.
07	`get_project`	Fetches specific details about one Supabase project using its unique reference ID.
08	`list_branches`	Retrieves a list of all development branches and their current status for a given project.
09	`list_organizations`	Shows all organizations linked to the account, along with their names and creation dates.
10	`list_projects`	Lists every Supabase project associated with the account, providing key info like region and status.
11	`list_secrets`	Displays all stored API keys and credentials for a specific project.
12	`pause_project`	Stops compute resources on a project, preserving the database data so costs stop accruing.
13	`restore_project`	Brings a previously paused Supabase project back to an active, fully operational state.

#	TOOL	DESCRIPTION
14	update_postgres_config	Modifies database settings like memory limits or connection pools for the project.

See It in Action

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

U Show me all my Supabase projects and their regions.

I found 4 projects: 'prod-api' (us-east-1, active), 'staging-api' (us-east-1, active), 'dev-sandbox' (eu-west-2, active) and 'analytics-db' (ap-southeast-1, paused). Would you like details on any specific project?

U Create a new dev branch called 'feature-auth' for my staging-api project.

Done! I've created the branch 'feature-auth' for your staging-api project. It has an isolated copy of the database that you can modify independently. The branch status is 'CREATING' and should be ready in a few minutes.

U Pause my dev-sandbox project to save costs.

Done! I've paused the 'dev-sandbox' project. All compute resources have been stopped and costs will no longer accrue. Your database and data are fully preserved. You can restore it anytime with `restore_project`.

Frequently Asked Questions

01 **How do I check project status with Supabase Alternative MCP?**

You use `list_projects` to see a comprehensive list of every associated project. This tool returns key details like the region and whether the project is active or paused.

02 **Can I change database settings using Supabase Alternative MCP?**

Yes, you can modify Postgres configuration parameters. Use `update_postgres_config` by providing the specific project reference and the JSON object containing the desired changes.

03 **What is the best way to test a new feature branch?**

First, use `create_branch`. This tool makes an isolated copy of your current database, allowing you and your team to experiment freely without any risk to the live project data.

04 **Is it safe to manage secrets with Supabase Alternative MCP?**

Yes. You use `create_secret` and `list_secrets`, and all credentials are encrypted at rest. Your agent handles this secure process, so you don't have to worry about manual key management.

05 **How do I stop paying for a project I'm not using?**

Run `pause_project` with the specific project reference. This preserves your database and data while stopping compute resources, preventing any further charges.

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 `"supabase-alternative": { "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

Supabase 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 Supabase. 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	Supabase MCP
Server ID	`019d8486-c486-711c-b7a0-75058a43806e`
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/supabase-alternative.