MCP SERVER | NO CODE | CLOUD HOSTED

Jestor MCP

Manage internal data, records, and automation flows.

Jestor lets your AI client manage complex internal data structures, workflows, and user records. It connects to a low-code API, allowing agents to list available datasets, retrieve specific entries, audit automated processes, and check system permissions for organizational databases.

A+ Quality Score 100/100

low-code | workflow-automation | database-management | internal-tools | task-management | data-modeling

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

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.

Jestor MCP

10 tools available
Cloud-hosted on Vinkius

Need your agent to interact with proprietary company data? This MCP gives it the ability to act like a skilled internal operations analyst. Instead of manually jumping through dashboards or opening multiple tabs just to pull a single piece of information, your AI client talks directly to Jestor's backend. You can ask it to list all available datasets, then retrieve specific records from those tables, and even audit every automated workflow running in the background. It's built for organizations that run on internal tools, making data access predictable and repeatable. Connecting through Vinkius means you don't have to manage ten different vendor connections; you just connect once from your preferred agent and get Jestor alongside everything else. This lets your team move past basic queries and actually perform deep database management tasks using simple natural language prompts.

Core Capabilities

01 — Identify available datasets

List all the data tables (objects) in the system so you know what information exists.

02 — Inspect data structure details

Get a schema for any object, telling you exactly what fields and relationships are available in that table.

03 — Fetch specific records by ID

Deep-dive into a single record or entry within any monitored dataset.

04 — Audit system processes

List and check the status of automated workflows, installed applications, and configured webhooks.

One Click on Vinkius — From Prompt to Execution

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

01 Your agent first uses `get_me` to confirm its connection status and view current user permissions.

02 It then uses `list_objects` to generate a catalog of all available data tables, guiding the next action.

03 Finally, it runs `list_records` or `get_record`, passing in the specific object name and necessary parameters to pull the required data.

The bottom line is that your agent treats your internal database like a structured API endpoint, allowing for precise, targeted data retrieval without manual UI interaction.

Built For

Data Operations Engineers and Business Analysts need this. If you spend half your day just finding out *where* the right data lives, you'll love Jestor. It’s for anyone whose job relies on querying complex, siloed internal systems.

Data Operations Engineer

Uses this to automate audits by running `list_workflows` and checking connectivity using `list_webhooks`, ensuring system reliability.

Business Analyst

Relies on it to quickly understand data relationships. They can use `get_object` to verify field types before building reports or making recommendations.

System Administrator

Uses the MCP to manage user accounts and system scope by calling `list_users`, which keeps track of who has access to what data.

What Changes When You Connect

01 You instantly know what data you can access. Instead of guessing where information lives, your agent calls `list_objects` to get a definitive list of all available datasets.

02 Audit your system logic without clicking around. Use `list_workflows` or `list_webhooks` to see exactly which automated processes are running and how they connect to external tools.

03 Get full context on data structures before querying. The `get_object` tool tells you the schema, so you know if a field is a date or text *before* your agent tries to read it.

04 Eliminate repetitive lookups for people. If you need to check who owns a record or find an employee's details, running `list_users` gives that data immediately.

05 Move beyond simple querying into deep auditing. You can retrieve specific records using `get_record`, and then cross-reference the owner ID against the user list via `list_users`.

Real-World Applications

Investigating a data discrepancy

A business analyst notices a client record is wrong. Instead of emailing three different department heads, they ask their agent to run `get_object` first. This verifies the schema, then they use `get_record` to pull the actual data and see exactly which fields are populated.

Onboarding a new team member

The system admin needs to verify who has access to sensitive client lists. They ask their agent to run `list_users` and then use `get_me` to confirm the scope of the current user's permissions, ensuring proper role assignment.

Debugging an automated process failure

A data ops engineer finds that a nightly report failed. They instruct their agent to run `list_workflows` to see which processes are scheduled, and then use `list_webhooks` to check if the necessary external connection points are still active.

Understanding application scope

A consultant needs a full picture of the platform. They ask their agent to run `list_apps` and `list_dashboards`. This gives them an immediate, comprehensive view of everything installed and visible to the end-user.

Patterns to Avoid

Assuming data existence

✗ AVOID

Telling your agent: 'Get me all client records from the Marketing table.' If that table doesn't exist or is named differently, the request fails and you waste time troubleshooting.

✓ INSTEAD

First, always run `list_objects` to confirm the exact name of the dataset. Then, use `list_records` with the correct object name to browse the data safely.

Confusing a list view with raw data

✗ AVOID

Asking for 'the client data' when the system only stores metadata. This leads to vague results because you can't pull the actual file content, just the record pointer.

✓ INSTEAD

If you want all clients, run `list_records` after confirming the object name. If you need field details, use `get_object`. Don't confuse the two.

Ignoring system dependencies

✗ AVOID

Trying to diagnose why a workflow failed without checking its connections. You might spend hours debugging data when the issue is just that an external endpoint changed.

✓ INSTEAD

Always check `list_webhooks` and run `list_workflows`. These tools show you the full operational map before you start troubleshooting specific record failures.

The Right Fit

Use this MCP if your primary need is interacting with structured, internal databases that are governed by low-code workflows. If you need to list all available datasets or audit who owns a process (using `list_objects`, `list_workflows`, or `list_users`), Jestor is the right tool. However, don't use this if your goal is general data transformation like cleaning messy CSVs; for that, look at specialized ETL connectors. Also, if you simply need to perform calculations on numbers without pulling them from an existing record, a dedicated calculation engine might be better than trying to pull all records via `list_records` just to crunch the numbers.

The administrative overhead of knowing where data lives

Right now, figuring out what data exists means opening a dozen different system dashboards. You jump from the 'CRM' tab to check client details, then open the 'Invoices' module just for billing history, and finally check the 'Marketing' dashboard to see who signed up last week. It's a manual process of clicking between tabs and copying identifiers.

With this MCP, your agent handles the discovery phase. You simply ask it what data is available, and using `list_objects`, it gives you an immediate map of every dataset—CRM, Invoices, Marketing—so you know exactly where to look without ever leaving the chat window.

Using Jestor for deep operational control

Manually checking a process failure requires multiple steps: first, finding the workflow name via `list_workflows`, then manually navigating to its history logs, and finally cross-referencing that against the user who triggered it using `list_users`. It's slow, error-prone detective work.

Now, your agent handles the entire audit trail. You can ask it to check a workflow's status and instantly get confirmation of all related webhooks or applications involved in the process. The complexity is managed by the API call.

Jestor: 10 Tools for Database Management

These tools let you programmatically interact with every aspect of your Jestor account, from listing datasets to auditing automated workflows.

#	TOOL	DESCRIPTION
01	`get_me`	Verifies your connection status and retrieves details about the currently authenticated user.
02	`get_object`	Retrieves the detailed schema, including field types and relationships for any specific data table.
03	`get_record`	Pulls all details for a single, specified entry or record in your database.
04	`list_apps`	Provides a list of every installed internal application available to the user.
05	`list_dashboards`	Lists all configured data visualization dashboards within the system.
06	`list_objects`	Returns a comprehensive list of every available dataset or object name in your account.
07	`list_records`	Lists all records belonging to a specific data table, allowing for browsing an entire dataset.
08	`list_users`	Retrieves names, emails, and unique IDs for every user in the organization's directory.
09	`list_webhooks`	Lists all external integrations configured via webhooks, useful for auditing third-party connections.
10	`list_workflows`	Retrieves a list of all automated workflows and event-driven business logic running in the background.

See It in Action

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

U List all objects in my Jestor account.

I'll fetch the list of all tables and objects for you.

U Show me the records for the 'Invoices' object.

I'll retrieve the records from your Invoices table in Jestor.

U Check the status of my workflows.

I'll look up the list of configured workflows and their status in Jestor.

Frequently Asked Questions

01 **How do I find out what data tables are available using Jestor MCP?**

Run `list_objects`. This tool immediately returns a comprehensive list of every object or dataset name in your account, giving you a clear scope.

02 **Can Jestor MCP help me audit my automated systems?**

Yes. You can use `list_workflows` to see all running processes and `list_webhooks` to check which external services are connected or configured for event triggers.

03 **What is the difference between getting a record and listing records with Jestor MCP?**

Use `get_record` when you know the specific ID of one item (like Client #123) and need its full details. Use `list_records` when you want to browse or see multiple items from an entire dataset.

04 **Does Jestor MCP tell me what fields are in a table?**

Yes, use the `get_object` tool. This fetches the detailed schema for any object, showing you every field name and its data type (text, date, number).

05 **How do I check user permissions with Jestor MCP?**

The `get_me` tool verifies your connection status and provides details about the currently authenticated user's profile and access rights.

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 `"jestor": { "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

Jestor 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 Jestor. 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	Jestor MCP
Server ID	`019d75bc-dfa9-7319-8da7-9bb22f61fa12`
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/jestor.