MCP SERVER | NO CODE | CLOUD HOSTED

Builder.io MCP for AI Agents

Organize and manage digital experience content models

Builder.io connects your AI agent directly to your visual CMS backend. It lets you manage content entries, check model schemas, and track reusable digital components without needing to log into a dashboard. You can list all content for specific models, update records programmatically, or even monitor API usage statistics—all through natural conversation.

A Quality Score 90/100

visual-cms | content-modeling | digital-experience | web-development | content-automation | component-management

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

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.

Builder.io MCP

10 tools available
Cloud-hosted on Vinkius

You need to manage complex web content, but logging into the CMS dashboard just to check a single page entry is slow. This MCP lets your AI agent bypass the UI entirely. Instead of manually navigating content models and checking schemas, you talk to it directly. You can ask your agent to list all blog posts or retrieve specific metadata from an old page template by name or ID. It handles everything from listing available spaces across your organization to updating a full content entry with new text and images. Because it's hosted on Vinkius, you connect once from any compatible client and get access to this powerful CMS tool alongside hundreds of others. You simply tell the agent what content you need—whether it's reviewing model structures or creating an entire set of new announcements—and it executes the required API calls in plain language.

Core Capabilities

01 — Retrieve Specific Content Details

Get detailed data for a single content entry or model type by its ID, allowing deep dives into the structure.

02 — Maintain CMS Structure and Assets

List all spaces, models, and reusable symbols to audit and understand the full scope of your digital architecture.

03 — Execute Content Changes

Create new content entries or update existing ones programmatically, treating the CMS like a data layer rather than a visual editor.

One Click on Vinkius — From Prompt to Execution

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

01 First, subscribe to this MCP and provide your Builder.io Public and Private API Keys.

02 Next, connect your AI agent or preferred client (like Cursor or Claude) to Vinkius using those credentials.

03 Finally, give your AI agent a natural language prompt—for example, 'What are the available content models?'—and it executes the necessary calls.

The bottom line is that you treat your entire CMS backend like an extension of conversation; no dashboards or manual clicks required.

Built For

Anyone who touches content structure, web development, or digital publishing will use this. It's for the engineers tired of boilerplate API calls and the editors who need to audit schemas quickly.

Web Developer

Uses the MCP to check content structures or update entries programmatically, saving time compared to writing manual API scripts.

Content Manager

Retrieves published content details and monitors model schemas directly from their workflow tools instead of switching tabs.

Digital Experience Architect

Orchestrates cross-space content updates and tracks overall API usage to plan complex digital campaigns.

What Changes When You Connect

01 Need to update a page? Use `update_content_entry` to modify published content instantly, bypassing the manual CMS editor.

02 Building out a new section? Run `list_models` to see every available content template right from your chat window. You get an immediate blueprint of what's possible.

03 Worrying about API limits? Check usage with `get_api_usage`. This tool provides real-time statistics so you never hit a ceiling unexpectedly.

04 Don't know the structure? Use `list_symbols` to see all reusable components, ensuring consistency across every part of your site architecture.

05 Getting specific data? Instead of searching dashboards, use `get_content_entry` or `get_model` to pull metadata and schema details instantly.

Real-World Applications

Reviewing a new website section

A developer needs to know what content types are available. They ask their agent, 'Show me all the possible models.' The agent uses `list_models` and returns a list of templates like 'Testimonial' and 'Gallery,' letting them start coding immediately.

Updating a seasonal announcement

The marketing team needs to change a banner across 20 pages. They ask the agent to update the specific entry ID for the announcement bar using `update_content_entry`. It confirms success instantly, saving hours of manual work.

Auditing content consistency

An architect wants to ensure all blog posts follow a strict schema. They ask the agent to use `get_model` on 'Blog Post' and review the required fields, ensuring every new article gets the necessary author reference.

Tracking organizational growth

A digital lead needs to see what spaces exist across departments. They ask their agent to run `list_spaces`. The result shows all 15 separate department workspaces, helping them route content management requests correctly.

Patterns to Avoid

Treating the CMS like a simple key-value store

✗ AVOID

Trying to update an entry just by knowing its title. You run `update_content_entry` but don't provide the necessary unique ID, and the request fails because it can't find the right record.

✓ INSTEAD

Always start by using `get_content_entry` or `list_content` to confirm the exact content entry ID or model details. This guarantees your updates are targeted precisely.

Overlooking component availability

✗ AVOID

Coding a new page section and realizing it needs an image gallery, but having no idea if that component is available in Builder.io's library.

✓ INSTEAD

Before building, run `list_symbols`. This tool provides the definitive list of all reusable components so you can build with confidence.

Ignoring schema changes

✗ AVOID

A developer assumes a content model ('Product') still has an 'SKU' field when they try to update it, but the CMS model was recently changed.

✓ INSTEAD

Run `get_model` first. It shows the absolute latest schema and metadata for that model, preventing code breaks due to outdated assumptions.

The Right Fit

Use this MCP when your workflow requires interacting with structured digital content—for example, listing articles or changing product details—without opening a web browser. This is ideal if you need to perform specific actions like `get_content_entry` or `list_models`. Don't use it if you only want general website browsing or simple text generation; for those tasks, a standard LLM connection works fine. You should avoid relying on this MCP if your primary goal is simply viewing the visual output of the CMS; you need to talk to its underlying structure and data layer. If you are just monitoring traffic, look into different analytics platforms instead.

Programmatically Managing Content Entries with Builder.io

Right now, updating a single page or auditing content requires logging into the CMS dashboard, clicking through multiple tabs (Models, Content Entries, Symbols), and manually checking IDs. It's slow, tedious, and makes it easy to mix up which version of the site you're actually working on.

With this MCP, the process changes completely. Instead of clicks, you ask your agent: 'Update the hero banner image for the homepage.' The system handles the `update_content_entry` call in the background and gives you a simple confirmation. You get back immediate, reliable control over your entire content layer.

Auditing CMS Architecture with Builder.io's Model Structure

Before this MCP, figuring out what fields a content model used—say, the 'Team Member' profile—required meant digging deep into the documentation or asking an engineer to share screenshots of the schema. It was guesswork based on old builds.

Now, you just ask your agent to run `get_model` for any template. It spits out the full schema metadata immediately. You know exactly what fields exist and what data type they expect before writing a single line of code.

Builder.io: 10 Tools for Content Model Management

Use these tools to create, retrieve, modify, and audit all the structured data within your Builder.io CMS backend via natural language commands.

#	TOOL	DESCRIPTION
01	`create_content_entry`	Creates a brand new piece of structured content in the CMS.
02	`get_content_entry`	Retrieves all detailed data for one specific content entry by its ID or name.
03	`get_model`	Shows the full schema structure and metadata for a single, specified content model type.
04	`get_space`	Gets detailed information about a specific workspace or container within your account.
05	`get_api_usage`	Pulls real-time statistics showing how much the CMS API has been used over time.
06	`list_content`	Lists all available content entries for a particular model type within your space.
07	`list_models`	Returns a list of every content model (like 'Page' or 'Blog') defined in the current workspace.
08	`list_spaces`	Provides an overview of all distinct workspaces available across your entire organization account.
09	`list_symbols`	Lists every reusable component or symbol that can be used across different pages and models.
10	`update_content_entry`	Modifies an existing piece of content, changing text, images, or metadata programmatically.

See It in Action

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

U What content models do I have available in my space?

Available Models:

- **Page**: Primary landing pages.
- **Blog Post**: Standard articles with author and date fields.
- **Testimonial**: Simple single-field component for quotes.

Please specify the model name if you want to list entries or check its schema.

U I need an update on the 'About Us' page. Can you pull up the current details?

Found it! Here are the details for content entry `page_3` :

- **Title:** About Our Company
- **Status:** Published
- **Last Modified:** 2026-05-15
- **Primary Model:** Page
- **Key Fields:** Mission Statement (Rich Text), Team Section (Reference to 'Team Member' model).

U Show me the structure for a blog post entry.

Here is the schema for the `blog-post` model:

FIELD NAME	DATA TYPE	REQUIRED
title	Text	Yes
content	Rich Text	Yes
author	Reference (User)	Yes
publishDate	Date	No

The `content` field accepts markdown and rich text, allowing for complex formatting.

Frequently Asked Questions

01 **How do I use the Builder.io MCP to manage content entries?**

You simply ask your AI agent what you want to do with the content (like 'List all blog posts'). The agent handles connecting to the CMS and executing the necessary calls, giving you structured data right in the chat.

02 **Can this MCP help me audit my digital experience models?**

Yes. It allows you to pull model schemas using the `get_model` tool. This means you can see exactly what fields every content type requires, helping you maintain consistency across your entire site.

03 **I need to update a page on my website; is this MCP right for me?**

Absolutely. You use the `update_content_entry` tool. You just give the agent the ID and the new data, and it performs the change directly in your CMS backend.

04 **Does Builder.io MCP let me see all available components?**

Yes. The `list_symbols` tool retrieves a list of every reusable symbol or component you've built into your CMS, which is crucial for maintaining design consistency.

05 **What if I need to know what spaces exist in my organization?**

Use the MCP to run `list_spaces`. This tool provides a complete overview of every workspace or environment you manage, so you can direct your content efforts to the right location.

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 `"builderio": { "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

Builder.io 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 Builder.io. 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	Builder.io MCP
Server ID	`019d7565-a1cc-70d0-b698-de620591c464`
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/builderio.