MCP SERVER | NO CODE | CLOUD HOSTED

XP Curve Generator MCP

Model Math for Character Progression Curves

The XP Curve Generator simulates game progression curves and analyzes player time investment for level design. It lets you model everything from basic linear growth to complex exponential requirements, giving you a precise breakdown of how much effort players will put into reaching specific levels or milestones.

B Quality Score 85/100

game-dev | xp-curve | leveling | balancing | progression

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · JUNE 2026
vinkius.com/mcp/xp-curve-generator

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.

XP Curve Generator MCP

0 tools available
Cloud-hosted on Vinkius

You're designing a new RPG, but figuring out the right XP curve is a nightmare. You need to hit certain difficulty points without making the final act feel impossible. This MCP lets you model character growth mathematically before writing a line of code. By feeding in your total desired experience and number of levels, your agent generates detailed tables showing exactly what requirements players will face at every stage. It identifies potential difficulty spikes—the 'walls' that might frustrate players—and helps you compare how different mathematical models affect the overall time commitment. If you already use Vinkius as your central catalog for other tools, adding this MCP gives you a deep math utility right alongside your game logic services.

Core Capabilities

01 — Calculate Level Requirements

Generate detailed level-by-level breakdowns of XP requirements and estimated play time based on chosen growth models.

02 — Identify Difficulty Spikes

Scan your progression data to locate abrupt difficulty increases or 'walls' that could frustrate players.

03 — Compare Curve Models

Evaluate how different scaling models, like linear versus quadratic curves, impact the total time a player spends in the game.

One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/mcp/xp-curve-generator** — connect your AI agent in three steps.

01 Specify your parameters: input the desired total XP, number of levels, and the growth model (e.g., Exponential or Quadratic).

02 The MCP runs simulations to map out progression data across various scaling models.

03 You receive detailed tables showing level-by-level XP requirements, estimated hours per level, and a comparison of the overall player time investment.

The bottom line is that you get quantifiable evidence about how your chosen math model will actually make players feel when they hit certain levels.

Built For

This MCP is for the Game Designer, the Systems Engineer, and the Balance Analyst. If your job involves predicting player retention or ensuring the final boss isn't a slog, you need this. Stop guessing at curves; start modeling them.

Game Designer

Uses this MCP to establish the fundamental mathematical backbone of character progression and level difficulty.

Level Designer

Checks if a specific zone's required XP fits logically into the overall game curve, preventing sudden drops or spikes in challenge.

Game Balance Engineer

Compares different scaling models to ensure that player time investment feels proportional across all major content arcs.

What Changes When You Connect

01 Pinpoint difficulty walls before playtesting. Use analyze_progression_spikes to identify exactly where players might get frustrated, saving massive amounts of rework later.

02 Compare Linear versus Quadratic models instantly using compare_scaling_models. Know which growth curve gives the best player retention without making the mid-game boring.

03 Get a complete level map with generate_progression_table. You'll see not just the XP numbers, but estimated hours per level, giving you solid data for content planning.

04 Stop guessing at math. This MCP lets you run simulations that tell you precisely how long a player will spend in the game under different mathematical constraints.

05 Manage your entire game development lifecycle from one place. When connected via Vinkius, this utility sits right next to your other core game logic services.

Real-World Applications

The Early Game Feels Too Slow

A team was worried the early levels were too easy and lacked challenge. They used compare_scaling_models, comparing a flat vs. exponential curve for 50 levels. The results showed that shifting to an exponential model increased player time investment by nearly 60% compared to their original linear plan.

Finding the Mid-Game Difficulty Wall

A level designer had a massive chunk of content in the mid-game, but wasn't sure if it was challenging enough. Running analyze_progression_spikes on their current data showed a significant difficulty wall at Level 28, allowing them to adjust rewards and pacing immediately.

Revising End-Game Requirements

The team needed to hit exactly 1,000,000 total XP over 50 levels. They used generate_progression_table with the desired parameters and adjusted their curve model until the final level jump felt satisfyingly large but achievable.

Justifying Scope Creep to Producers

The game balance team needed hard data to justify adding a whole new system. They used compare_scaling_models, showing that switching from a quadratic to a linear model would actually reduce the overall player time investment too much.

Patterns to Avoid

Using Spreadsheets for Curve Design

✗ AVOID

Manually adjusting cells in Excel based on feeling. This approach is slow, prone to calculation errors, and doesn't account for the overall player experience.

✓ INSTEAD

Run simulations with generate_progression_table. Input your total XP and level count, then let the MCP calculate the requirements mathematically, ensuring consistency across all levels.

Only Checking Final Level Difficulty

✗ AVOID

Focusing only on the final boss's required XP without considering what happens in the preceding 80% of the game. This often leads to massive difficulty spikes or boredom.

✓ INSTEAD

Use analyze_progression_spikes immediately after generating your data set. It flags potential 'difficulty walls' across the entire spectrum, not just at the end.

Ignoring Model Comparison

✗ AVOID

Picking a curve model (e.g., Quadratic) without comparing it to alternatives like Linear or Flat Early/Steep Late. You might pick one that kills player motivation too early.

✓ INSTEAD

Always run compare_scaling_models. This shows you the time difference and impact of swapping models, letting you choose the curve that best matches your pacing goals.

The Right Fit

Use this MCP if your problem is mathematical: you need to determine *what* progression curve should exist to hit a specific target (e.g., 1 million XP in 50 levels). You need to know the quantitative trade-offs between models like linear versus exponential growth.

Don't use it if your problem is qualitative or content-based; for instance, if you just need to track how many items a player has gathered, that requires an inventory tracking tool. Similarly, if you only need simple stat calculation (like total damage output), basic calculators suffice. This MCP is strictly about mathematical progression modeling and timing.

The Spreadsheet Nightmare of Game Balancing

Right now, game balance often means spending days in spreadsheets, manually adjusting formulas to predict how many hours a player might spend reaching Level 35. You calculate the XP requirements for one level, then have to copy that formula down hundreds of rows, hoping your totals add up and that you haven't accidentally introduced a massive spike at the worst possible moment.

With this MCP, you define the boundaries—the total XP and the number of levels—and let your agent do the heavy lifting. You get detailed progression tables instantly, allowing you to visualize every level jump and know exactly what curve keeps players engaged until the credits roll.

XP Curve Generator: Predict Player Effort

You no longer have to manually calculate how much time difference there is between a linear progression versus an exponential one. Running compare_scaling_models gives you the exact hours saved or lost by switching models, letting your team argue with hard numbers instead of gut feelings.

This MCP moves curve design from tedious calculation into rapid analysis. You move from guesswork to concrete data about player pacing.

See It in Action

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

U Generate an XP curve for a 50 level system with 1,000,000 total XP using an exponential curve and 5000 XP per hour.

The progression table has been generated. Level 1 requires 0 XP to reach, with the next level requiring approximately 52.63 XP, and the final level jump is significantly larger.

U Check if there are any major difficulty spikes in my current progression data.

No significant spikes were detected above the 50% threshold in the provided progression data.

U Compare a linear curve vs a quadratic curve for 100 levels and 500,000 XP at 1000 XP/hour.

The Linear model results in a faster progression with a time difference of approximately 45.2 hours compared to the Quadratic model.

Frequently Asked Questions

01 **What does the XP Curve Generator MCP use if I only want a simple linear increase?**

You specify 'Linear' as your growth model when using generate_progression_table. The tool will calculate consistent, steady increases for every level requirement.

02 **How can I check for difficulty spikes with the XP Curve Generator MCP?**

Use analyze_progression_spikes. You feed it your current progression data, and it flags any sudden jumps above a certain threshold that might frustrate players.

03 **Can the XP Curve Generator compare different types of curves?**

Yes, use compare_scaling_models to evaluate models like Quadratic versus Exponential. It provides a direct comparison of their impact on total player time investment.

04 **Does this MCP account for variable playtime?**

It estimates hours per level based on your input rate (e.g., 5000 XP per hour). These estimations help you adjust the difficulty curve to match desired player pacing.

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 `"xp-curve-generator": { "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

XP Curve Generator 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 XP Curve Generator. 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	XP Curve Generator MCP
Server ID	`019efdb7-76e2-72e4-9088-25dee0475c79`
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/xp-curve-generator.