Changelog

Unreleased (feat/phase1-baseline)

Recovery Notes (August 2025)

- ✅ **Complete System Recovery**: Successfully recovered from git revert data loss incident
- 🔍 **Forensic Analysis**: Identified and preserved latest UI version (1577 lines) and evolved backend (1778 lines)
- 🧹 **Workspace Cleanup**: Organized recovery materials and removed failed modular upgrade remnants
- ✅ **Full Verification**: 49/49 test suite passing, all MCP server connections operational
- 🚀 **Production Ready**: System fully restored and verified working with all 4 MCP servers (Management, Perplexity, Time, Playwright)

Added

- 🎛 **Real-time Log Monitoring & UI Management**: Complete management interface at `/ui` with live server logs, configuration editing, and comprehensive tool visibility
- 🔧 **Dynamic Configuration Management**: Edit `mcpo.json` directly through the UI with real-time validation, backup creation, and instant server reload
- 📦 **Python Dependency Management**: UI-based `requirements.txt` editing with automatic `pip install` on server startup and reload
- 🛠 **Self-Managed Internal Tools**: MCPO now exposes its own management capabilities as discoverable MCP tools at `/mcpo/openapi.json`
  - `install_python_package`: Dynamically install Python packages via pip
  - `get_config`: Retrieve current `mcpo.json` configuration
  - `post_config`: Update configuration and trigger server reload
  - `get_logs`: Retrieve last 20 server log entries
- 📊 **Enhanced Log Buffer System**: In-memory log capture with thread-safe access for real-time UI display via `/_meta/logs`
- 🔄 **Advanced Configuration Endpoints**: Complete CRUD operations for configuration management
  - `/_meta/config/content`: Get formatted `mcpo.json` content
  - `/_meta/config/save`: Save and validate configuration with backup and reload
  - `/_meta/requirements/content`: Get `requirements.txt` content
  - `/_meta/requirements/save`: Save dependencies and trigger installation
- 🎨 **Modern Configuration UI**: Single-page interface with tabs for servers, logs, and configuration editing
- 🔍 **Tool Call Execution Logging**: Comprehensive logging of MCP tool calls with success/failure tracking and detailed error reporting
- 🏗 **Internal MCP Server Architecture**: Self-hosted tools using mock MCP sessions for seamless integration with existing tool discovery
- `/healthz` endpoint and health snapshot
- Pydantic config models (`AppConfig`, `ServerConfig`)

- Global reload lock for atomic config reloads
- `--tool-timeout` CLI option (default 30s)
- `--structured-output` flag (experimental) adding typed collection envelope
- Enum and min/max length/number constraint exposure in dynamic models
- MCP protocol version header injection (`MCP-Protocol-Version: 2025-06-18`)
- Unified success + error envelope helpers and global HTTP/validation handlers
- Tool-level unified error envelope (consistent `{ok:false,error:{...}}` shape)
- Tests for structured output and tool error envelope
- Enforced per-invocation tool timeout with async cancellation (`asyncio.wait_for`)
- Per-request timeout override via `X-Tool-Timeout` header or `?timeout=` query param
- Hard upper bound `--tool-timeout-max` with validation and error envelope
- Meta endpoints: `/_meta/servers`, `/_meta/servers/{server}/tools`, `/_meta/config`
- Dynamic config reload endpoint `/_meta/reload` and per-server reinit `/_meta/reinit/{server}`
- Server & tool enable/disable endpoints (403 enforcement for disabled tools)
- In-browser settings UI at `/mcp` with theme toggle, >40 tool warning, expandable server panels
- Add / remove server endpoints (config mode persistence) and modal (Git analysis stub + manual path)
- Open config action (vscode:// deep link) from UI
- `--read-only` flag to disable all mutating management endpoints for safer embedding/distribution
- Versioned, atomic state persistence for server/tool enable flags (`*_state.json` with temp-file replace)

Changed

- Correct README Python version to 3.11+
- Cleanup duplicate imports
- Removed unused JWT/passlib dependencies from default install (simplifies surface area; API key model only in phase 1)

Planned / Pending

- Expanded structured output (streaming, richer resource metadata)
- Additional tests for image/resource items & timeout behavior

---

Historical entries are maintained upstream; this fork annotates divergence points below.

[0.0.17] - 2025-07-22

Added

- 🔄 **Hot Reload Support for Configuration Files**: Added `--hot-reload` flag to watch your config file for changes and dynamically reload MCP servers without restarting the application—enabling seamless development workflows and runtime configuration updates.
- 🔇 **HTTP Request Filtering for Cleaner Logs**: Added configurable log filtering to reduce noise from frequent HTTP requests, making debugging and monitoring much clearer in production environments.

Changed

- ⬆️ **Updated MCP Package to v1.12.1**: Upgraded MCP dependency to resolve compatibility issues with Pydantic and improve overall stability and performance.
- 🔧 **Normalized Streamable HTTP Configuration**: Streamlined configuration syntax for streamable-http servers to align with MCP standards while maintaining backward compatibility.

[0.0.16] - 2025-07-02

Added

- 🔁 **Enhanced Endpoint Support for Arbitrary Return Types**: Endpoints can now return any JSON-serializable value—removing limitations on tool outputs and enabling support for more diverse workflows, including advanced data structures or dynamic return formats.
- 🪵 **Improved Log Clarity with Streamlined Print Trace Output**: Internal logging has been upgraded with more structured and interpretable print traces, giving users clearer visibility into backend tool behavior and execution flow—especially helpful when debugging multi-agent sequences or nested toolchains.

Fixed

- 🔄 **Resolved Infinite Loop Edge Case in Custom Schema Inference**: Fixed a bug where circular references ($ref) caused schema processing to hang or crash in rare custom schema setups—ensuring robust and reliable auto-documentation even for deeply nested models.

[0.0.15] - 2025-06-06

Added

- 🔐 **Support for Custom Headers in SSE and Streamable Http MCP Connections**: You can now pass custom HTTP headers (e.g., for authentication tokens or trace IDs) when connecting to SSE or streamable_http servers—enabling seamless integration with remote APIs that require secure or contextual headers.
- ▨ **MCP Server Instructions Exposure**: mcpo now detects and exposes instructions output by MCP tools, bringing descriptive setup guidelines and usage help directly into the OpenAPI schema—so users, UIs, and LLM agents can better understand tool capabilities with zero additional config.
- 🧪 **MCP Exception Stacktrace Printing During Failures**: When a connected MCP server raises an internal error, mcpo now displays the detailed stacktrace from the tool directly in the logs—making debugging on failure dramatically easier for developers and MLops teams working on complex flows.

Fixed

- 🧩 **Corrected Handling of Underscore Prefix Parameters in Pydantic Modes**: Parameters with leading underscores (e.g. _token) now work correctly without conflict or omission in auto-generated schemas—eliminating validation issues and improving compatibility with tools relying on such parameter naming conventions.

[0.0.14] - 2025-05-11

Added

- 🌐 **Streamable HTTP Transport Support**: mcpo now supports MCP servers using the Streamable HTTP transport. This allows for more flexible and robust communication, including session management and resumable streams. Configure via CLI with '--server-type "streamable_http" -- ' or in the config file with 'type: "streamable_http"' and a 'url'.

[0.0.13] - 2025-05-01

Added

- 🧪 **Support for Mixed and Union Types (anyOf/nullables)**: mcpo now accurately exposes OpenAPI schemas with anyOf compositions and nullable fields.
- 🔗 **Authentication-Required Docs Access with --strict-auth**: When enabled, the new --strict-auth option restricts access to both the tool endpoints and their interactive documentation pages—ensuring sensitive internal services aren’t inadvertently exposed to unauthenticated users or LLMs.
- 🧬 **Custom Schema Definitions for Complex Models**: Developers can now register custom BaseModel schemas with arbitrary nesting and field variants, allowing precise OpenAPI representations of deeply structured payloads—ensuring crystal-clear docs and compatibility for multi-layered data workflows.
- 🔄 **Smarter Schema Inference Across Data Types**: Schema generation has been enhanced to gracefully handle nested unions, nulls, and fallback types, dramatically improving accuracy in tools using variable output formats or flexible data contracts.

[0.0.12] - 2025-04-14

Fixed

- ⌛ **Disabled SSE Read Timeout to Prevent Inactivity Errors**: Resolved an issue where Server-Sent Events (SSE) MCP tools would unexpectedly terminate after 5 minutes of no activity—ensuring durable, always-on connections for real-time workflows like streaming updates, live dashboards, or long-running agents.

[0.0.11] - 2025-04-12

Added

- 🌊 **SSE-Based MCP Server Support**: mcpo now supports SSE (Server-Sent Events) MCP servers out of the box—just pass 'mcpo --server-type "sse" -- http://127.0.0.1:8001/sse' when launching or use the standard "url" field in your config for seamless real-time integration with streaming MCP endpoints; see the README for full examples and enhanced workflows with live progress, event pushes, and interactive updates.

[0.0.10] - 2025-04-10

Added

- 📦 **Support for --env-path to Load Environment Variables from File**: Use the new --env-path flag to securely pass environment variables via a .env-style file—making it easier than ever to manage secrets and config without cluttering your CLI or exposing sensitive data.
- 🧪 **Enhanced Support for Nested Object and Array Types in OpenAPI Schema**: Tools with complex input/output structures (e.g., JSON payloads with arrays or nested fields) are now correctly interpreted and exposed with accurate OpenAPI documentation—making form-based testing in the UI smoother and integrations far more predictable.
- 🛑 **Smart HTTP Exceptions for Better Debugging**: Clear, structured HTTP error responses are now automatically returned for bad requests or internal tool errors—helping users immediately understand what went wrong without digging through raw traces.

Fixed

- 🪛 **Fixed --env Flag Behavior for Inline Environment Variables**: Resolved issues where the --env CLI flag silently failed or misbehaved—environment injection is now consistent and reliable whether passed inline with --env or via --env-path.

[0.0.9] - 2025-04-06

Added

- 🧭 **Clearer Docs Navigation with Path Awareness**: Optimized the /docs and /[tool]/docs pages to clearly display full endpoint paths when using mcpo --config, making it obvious where each tool is hosted—no more guessing or confusion when running multiple tools under different routes.
- 🛣 **New --path-prefix Option for Precise Routing Control**: Introduced optional --path-prefix flag allowing you to customize the route prefix for all mounted tools—great for integrating mcpo into existing infrastructures, reverse proxies, or multi-service APIs without route collisions.
- 🐳 **Official Dockerfile for Easy Deployment**: Added a first-party Dockerfile so you can containerize mcpo in seconds—perfect for deploying to production, shipping models with standardized dependencies, and running anywhere with a consistent environment.

[0.0.8] - 2025-04-03

Added

- 🔒 **SSL Support via '--ssl-certfile' and '--ssl-keyfile'**: Easily enable HTTPS for your mcpo servers by passing certificate and key files—ideal for securing deployments in production, enabling encrypted communication between clients (e.g. browsers, AI agents) and your MCP tools without external proxies.

[0.0.7] - 2025-04-03

Added

- 🖼 **Image Content Output Support**: mcpo now gracefully handles image outputs from MCP tools —returning them directly as binary image content so users can render or download visuals instantly, unlocking powerful new use cases like dynamic charts, AI art, and diagnostics through any standard HTTP client or browser.

[0.0.6] - 2025-04-02

Added

- 🔐 **CLI Auth with --api-key**: Secure your endpoints effortlessly with the new --api-key option, enabling basic API key authentication for instant protection without custom middleware or external auth systems—ideal for public or multi-agent deployments.
- 🌐 **Flexible CORS Access via --cors-allow-origins**: Unlock controlled cross-origin access with the new --cors-allow-origins CLI flag—perfect for integrating mcpo with frontend apps, remote UIs, or cloud dashboards while maintaining CORS security.

Fixed

- 🖌 **Cleaner Proxy Output**: Dropped None arguments from proxy requests, resulting in reduced clutter and improved interoperability with servers expecting clean inputs—ensuring more reliable downstream performance with MCP tools.