AI MCP Server

VeloxFactory can be operated directly through an AI assistant. A dedicated **MCP Server**, available as an add-on for VeloxFactory customers, exposes report configs, rendering, print dispatching, mailing, recurring jobs and user administration through the Model Context Protocol (MCP), so any MCP-compatible AI assistant can manage VeloxFactory through plain conversation instead of the frontend or hand-written API calls.

> **In short:** if you can describe the task, your assistant can do it. No API documentation to read, no frontend clicks to remember, and not tied to any single AI provider.

What It Lets You Do

The MCP Server gives your assistant access to everything VeloxFactory can do, organised into a few practical areas.

Manage reports by talking, not clicking. Create, update, and organise report configs, connection configs, and report contexts through conversation. "Clone the KanBan label as a QA version and rename the fields" is a request your assistant can carry out directly.

Render and preview on demand. Ask for any report to be rendered with real or example data, and get a finished PDF back, opened automatically, ready to view. No need to log into the frontend or assemble an API request by hand. This includes swapping an image in on the fly for a single render, a file from the instance's resources folder, an `https://` URL, or a Base64 upload, without editing the report configuration itself.

Handle files without the round-trip. Uploading a new `.jrxml` template, swapping a logo, or attaching a resource image is as simple as pointing your assistant at the file on your computer. It takes care of the technical side.

Dispatch print jobs. Have a rendered document sent straight to a print task, check its status, or mark it as printed, the same dispatching VeloxFactory already does, just requested in a sentence. Printers can be named the way people name them: VeloxFactory resolves a display name such as *Warehouse Label 01* to the queue name of the print server and fills in the broadcast channel by itself, so the assistant never has to look a printer up first.

Maintain the printer master data. List, create, update, retire, or delete the printers your company prints on, including location, type and broadcast ID, so the pickers in the frontend and every print task stay in sync with reality.

Send a rendering by mail. A render can go out as a mail in the same request, with the PDF and an optional xlsx attached. Mailers and mail templates are referenced by name, so "*render the delivery note for order 4711 and mail it to dispatch*" is one message. The assistant can also set the mail side up from scratch: create an SMTP account from a provider's documented settings, send a test mail to prove the credentials, write a template against the real placeholder catalogue of a specific report, and preview it before anyone receives it. A document that was rendered earlier can be mailed again at any time, to a freely chosen recipient.

Put recurring work on a schedule. The shift report that belongs in a mailbox at six every morning does not need anyone to ask for it. The assistant can turn a sentence into a schedule, confirm the crontab expression by reading the next run times back, run the schedule once by hand to check it, and only then switch it on. A schedule can just as well be pointed at a single date and time instead of a crontab expression, firing exactly once and, if wanted, deactivating itself afterwards. The cleanup schedule that keeps history, print tasks, mail tasks and files from growing without bound is maintained the same way, one retention at a time.

Administer users and access. Create accounts, adjust permissions, reset passwords and enable or disable access. Day-to-day administration happens in conversation instead of in the interface.

Keep track of history. Pull up past renderings, review what was generated when and by whom, follow a trace id from a scheduled run through to the mail it produced, and clean up records that are no longer needed.

At a Glance

Coverage is complete, every part of VeloxFactory is reachable, not just the rendering side:

Area	What's covered
Users	List, view, create and update accounts; change passwords; enable and disable access
Report Contexts	List, view, create, update, and delete
Connection Configs	List, view, create, update, delete, and test a database connection
Common Report Resources	List, view, create, update, and delete (logos, images, shared files)
Report Configs	List, view, create, update, delete, manage resources/parameters/fields, generate previews, and render (including per-render image overrides via `resourceOverrides` and an optional `mailing` segment)

Area	What's covered
Report Resources	Link or unlink a Common Report Resource to a report
History Records	List, view, create, update, delete, print, mail again, and clean up linked print tasks
Print Tasks	List, view, create, update, delete, and set status/printed state
Printers	List, view, create, update, and delete printer master data; names are resolved automatically when a print task is dispatched
Mailers	List, view, create, update, and delete SMTP accounts, and send a test mail to prove the credentials
Mail Templates	List, view, create, update, and delete; read the placeholder catalogue of a report, and preview subject and body without sending anything
Mail Queue	List and view mail tasks, mail an existing rendering, create a task directly, repeat one with corrected recipients, and delete
Scheduled Jobs	List, view, create, update, and delete schedules; validate a crontab expression and preview its next run times; set a schedule to run once at a given date and time instead of recurring; read the placeholders a schedule may use; run one immediately; adjust the retentions of the cleanup schedule
Run Log	List and view the runs of a schedule, searchable by trace id and outcome, and delete single entries
Local Utilities	Automatically open rendered PDFs, URLs, and output folders on your machine

89 tools in total, covering the full VeloxFactory API.

> ℹ **API tokens are not part of the tool set.** They are issued and revoked in the web interface only. The MCP server talks to VeloxFactory with a token of its own, and a request authenticated that way cannot manage tokens. Asked for a token, the assistant points at **Users → user → API Tokens** instead.

What This Looks Like in Practice

A few examples of the kind of request the MCP Server is built for:

- "Render the A5 KanBan label for article 4561287-154 and show me the PDF."
- "I designed a new label in Jaspersoft Studio, onboard it as a report config."
- "Create a new user for our warehouse lead with admin rights."
- "Print this delivery note to the shop floor printer and let me know once it's done."
- "Show me every report config connected to the QA database."
- "Render this label but use this product photo instead of the linked one, just for this one print."
- "Add the new label printer in hall 2 and send the next delivery note there."
- "Mail this morning's shift report to the supervisor again, this time with the Excel export."
- "Set up our Office 365 account as a mailer and send a test mail to check it works."
- "Send the weekly stock list to disposition every Monday at six, and tell me the next three times it would run."
- "Run the year-end archive job once on New Year's Eve at midnight and turn the schedule off afterwards."
- "Which scheduled runs failed last night, and what did they say?"

Each of these would otherwise mean a frontend session or a small API script. With the MCP Server, they are a single message.

Built-In Know-How

Access to the tools is only half the story. Alongside the MCP Server, kiwi software also provides a companion **Skill**, a packaged layer of accumulated know-how that teaches the assistant how to use those tools well, not just that they exist.

In practice, this is what separates "the assistant can technically call the API" from "the assistant makes the same good judgment calls a person who knows VeloxFactory inside out would make": choosing the right output format for the situation, batching multiple renders sensibly instead of opening five windows in a row, staying within an instance's rate limit, and following VeloxFactory's own conventions rather than generic assumptions.

It also covers the judgment calls that only matter once something runs unattended: checking what a render actually reported about its mail instead of trusting a 200, and testing a schedule by hand before giving it a cron expression, so nobody finds out at three in the morning that it prints four hundred labels.

The most visible example: with the Skill in place, the assistant can **design a brand-new report template from scratch** (a label, delivery note, or certificate) without ever opening Jaspersoft Studio. It knows the exact conventions a `.jrxml` file needs to work in VeloxFactory (parameter naming, example values, and so on) and checks its own work with a preview before calling the template done.

□□ The Skill ships together with the MCP Server as part of the same add-on, there is nothing separate to install or configure.

Who It Is For

The MCP Server is aimed at teams that already run VeloxFactory and want a faster way to handle the routine work around it, onboarding new report templates, day-to-day administration, one-off renderings, or quick checks on what has already been generated.

It is a natural fit for anyone comfortable working with an AI assistant who would rather describe a task than click through a frontend or write an API call, including teams without dedicated developer resources to spare on integration work. Since MCP is an open, provider-independent standard, it is not limited to a single AI product.

Getting It

The MCP Server is built and maintained by kiwi software as an add-on to a VeloxFactory license. It runs locally alongside your MCP-compatible AI assistant (Claude being one example) and connects to your VeloxFactory instance using your own API credentials. Nothing about your data passes through a third party.

Setup is handled together with kiwi software as part of onboarding, typically alongside the initial VeloxFactory rollout or added to an existing installation.

> ℹ **Interested?** Reach out via the Get VeloxFactory page, or try the live demo first at demo.veloxfactory.dev.

Revision #20
Created 2026-06-30 21:52:33 UTC by Benjamin Fischer
Updated 2026-09-23 05:55:02 UTC by Benjamin Fischer