GHOST LOGICS LLC

LM Studio
Beyond the Basics

Advanced Settings, Model Tuning & Workflow Tricks

A companion volume to the AI Dev Studio Field Manual.
Everything past the install-and-load basics.

Table of Contents

1. Choosing the Right Model: Format & Quantization
2. Context Window Configuration
3. GPU Offloading & Performance Tuning
4. Sampling Parameters Explained (With Examples)
5. Preset & System Prompt Management
6. Multi-Model Workflows
7. The Local Server — Advanced Settings
8. Performance Troubleshooting
9. Quick Reference Cheat Sheet

SECTION 01

Choosing the Right Model

Format & quantization — what you're actually trading off

Every model you download is offered in multiple “quantization” levels — compressed versions that trade precision for size and speed. Picking the right one matters more than picking the right model name.

Quantization Levels, Plain English

Level	What it means	When to use it
Q8 / FP16	Near-full precision, largest file, slowest	Only if you have ample RAM/VRAM and want maximum quality for a hard task
Q6_K	Very close to Q8 quality, notably smaller	Good middle ground if you have the RAM to spare
Q4_K_M	The sweet spot for most coding use — small quality loss, big size/speed win	Default recommendation for day-to-day coding work
Q3_K_M	Noticeable quality drop, much smaller	Only on constrained hardware, expect more mistakes
Q2_K	Significant quality loss	Avoid for coding tasks — fine only for the lightest autocomplete-style use

Rule of Thumb
For a 7B coding model, Q4_K_M needs roughly 4-5GB of RAM/VRAM and is very hard to tell apart from Q8 in day-to-day coding tasks. Don't reach for a bigger quantization by default — test Q4_K_M first and only go up if you actually notice quality problems.

GGUF: The Format You'll See Everywhere

GGUF is the standard file format LM Studio (and most local-inference tools) use. When browsing models, you'll usually see several GGUF files for the same model, one per quantization level — that's the choice described above, not a different model each time.

SECTION 02

Context Window Configuration

The context window is how much text (your prompt plus the conversation history plus any files you've shown it) the model can “see” at once. Get this wrong in either direction and quality suffers in ways that don't look like an obvious error.

What Happens When You Exceed It

Silent Degradation, Not a Clean Error
Exceeding context doesn't always throw an obvious error — often the model just starts “forgetting” earlier instructions or file contents, silently dropping the oldest parts of the conversation. If a local model suddenly seems to have forgotten something you told it several messages ago, check your context length setting before assuming it's a model quality problem.

Setting It Correctly

- In LM Studio's model loading settings, there's a context length slider/field — don't just leave it at whatever default loaded
- For single-file edits: a smaller context (4K-8K tokens) is fine and loads/runs faster
- For multi-file review or long debugging sessions: raise it (16K-32K+ if your model supports it and you have the RAM) — check the specific model's max supported context first, since going past what the model was actually trained/tuned for degrades quality even if the software allows it
- Higher context = more RAM used, even before you've filled it — the memory is reserved upfront, not allocated as you go

SECTION 03

GPU Offloading & Performance Tuning

GPU Offload Layers — What This Actually Means

A model is made of many computational “layers.” The GPU offload setting controls how many of those layers run on your graphics card versus your CPU. More layers on the GPU = faster generation, up to the point your GPU's VRAM runs out.

Practical Approach

1. Start by setting GPU offload to “max” / as high as LM Studio allows for the loaded model
2. If the model fails to load or your system becomes unstable, that's your VRAM ceiling — lower it in increments until it loads cleanly
3. Watch generation speed (tokens/sec, shown in LM Studio's chat interface) as you adjust — there's usually a clear jump in speed once enough layers are on GPU

Confirming the GPU Is Actually Being Used

- Open Windows Task Manager → Performance tab → GPU — generate a response and watch for GPU usage spiking
- If GPU usage stays near zero during generation, the model is running entirely on CPU regardless of your offload setting — check that you have the correct GPU-enabled build of LM Studio's runtime for your hardware (NVIDIA/CUDA, AMD/ROCm, or Apple Silicon/Metal each need their own matching backend)

CPU-Only Fallback

If your GPU doesn't have enough VRAM for a given model even at low offload, LM Studio can still run entirely on CPU — it'll work, just noticeably slower. For a 7B model at Q4, expect a real, felt difference between GPU-accelerated and CPU-only generation speed.

Flash Attention

If your loaded model and hardware support it, enabling Flash Attention (a toggle in the model loading settings) reduces memory use and can improve speed with no quality cost. Worth turning on by default and only disabling if you hit a compatibility error on load.

SECTION 04

Sampling Parameters Explained

With concrete before/after examples

These settings control how the model chooses its next word at each step. Getting them right for the task at hand matters more than most people realize — the same model can feel completely different in reliability depending on these values.

Temperature

Controls randomness. Lower = more predictable/deterministic, higher = more varied/creative.

Setting	Behavior	Use for
0.0 – 0.2	Highly deterministic, same-ish answer every time	DEV Deterministic tasks, exact bug fixes, code that must be predictable
0.3 – 0.5	Mostly consistent with a little variation	General coding, refactors
0.7 – 0.9	Noticeably more varied, more creative phrasing	Brainstorming, DEV Creative, Growth Hacker sessions
1.0+	High variety, higher risk of incoherence	Rarely useful for coding — mainly pure ideation volume

Concrete Example
Ask the same model to “suggest a name for a plant-care app feature” at temperature 0.1 three times — you'll get the same or nearly identical answer each time. Ask at temperature 0.9 three times — you'll get three genuinely different suggestions. Neither is “wrong” — they're suited to different jobs.

Top-P (Nucleus Sampling)

Restricts the model to choosing from only the most probable next-words that together make up the top P% of probability. Lower top-P (e.g. 0.85) tightens output further; 0.9-0.95 is a reasonable default for most coding work and rarely needs touching.

Repeat Penalty

Discourages the model from repeating the same phrase or getting stuck in a loop. If you notice a local model repeating itself verbatim in long responses, raising this slightly (try 1.1-1.15) usually fixes it. Setting it too high can make output feel unnaturally forced to avoid repetition, even when repetition would be correct (like repeating a variable name).

SECTION 05

Preset & System Prompt Management

You already have 11 named agent system prompts from the Field Manual. LM Studio lets you save each as a reusable preset so switching personas is one click, not a copy-paste each time.

Setting Up a Preset

1. In LM Studio's chat interface, open the system prompt field (usually a collapsible panel at the top or side of the chat)
2. Paste in the full agent prompt (e.g. DEV Deterministic)
3. Look for a “save preset” or “save as” option — name it clearly (e.g. “GL - DEV Deterministic”, using a consistent prefix so all 11 sort together)
4. Repeat for each of the 11 agents

Naming Convention Worth Adopting

```
GL - DEV Deterministic
GL - Refactor Surgeon
GL - Security Officer
GL - Architect
GL - Market Research
GL - Creative
GL - Growth Hacker
GL - Blue Ocean
GL - Monetization
GL - Founder Mode
GL - QA Engineer
```

A shared prefix keeps all 11 grouped together in the preset dropdown instead of scattered alphabetically among any other presets you create later.

Pairing Presets with Sampling Settings

Presets can often save more than just the system prompt — check whether your version of LM Studio lets a preset also remember its own temperature/top-p settings. If so, save each agent with its ideal temperature baked in (low for Deterministic/QA/Refactor Surgeon, higher for Creative/Growth Hacker/Blue Ocean) so you don't have to remember to adjust it each time you switch.

SECTION 06

Multi-Model Workflows

Running Two Models for Comparison

If your hardware allows it, some workflows benefit from asking the same question to two different local models and comparing answers — particularly useful when you don't have a paid model handy to sanity-check against, or want a free second opinion before spending a cloud credit on it.

Different Models for Different Roles, Even Locally

- A smaller, faster model (e.g. Qwen2.5-Coder 0.5B) for tab-autocomplete-style suggestions where speed matters more than depth
- A larger model (7B or bigger, if your hardware supports it) for actual chat/reasoning tasks
- This mirrors exactly how your Continue config already splits `local-coder` and `local-autocomplete` — the same principle applies inside LM Studio's own standalone chat too

Loading Multiple Models at Once

LM Studio can keep more than one model loaded simultaneously if you have the RAM for it, letting you switch between them instantly in the chat window without a reload delay. Useful if you regularly bounce between a fast small model and a slower capable one — costs you standing RAM usage in exchange for instant switching.

SECTION 07

The Local Server — Advanced Settings

This is the piece Continue actually talks to. A few settings beyond “just turn it on” are worth knowing.

Port Configuration

Default is 1234. If that port is ever in use by something else on your machine, LM Studio lets you change it in the server settings — just remember to update the matching `apiBase` URL in your Continue `config.yaml` to match, or Continue will silently fail to connect.

Request Logging

The local server tab typically shows a live log of incoming requests. This is genuinely useful for debugging — if Continue seems to be sending nothing, or sending malformed requests, this log tells you immediately rather than guessing.

Concurrency

Local inference is generally single-threaded per model — if Continue sends a second request while one's still processing, it typically queues rather than running in parallel. This matters if you're used to cloud models handling multiple simultaneous requests — expect local requests to serialize.

SECTION 08

Performance Troubleshooting

“Generation Is Very Slow” Checklist

1. Check GPU usage in Task Manager during generation — near-zero means you're on CPU fallback
2. Check how many layers are actually offloaded to GPU in the model settings
3. Check what else is competing for RAM/VRAM at the same time — a native Android/iOS build process running simultaneously is a known real cause of this on constrained hardware
4. Check the model's quantization level — a Q8 model will always be slower than the same model at Q4
5. Check context length isn't set far higher than what you're actually using — a huge reserved context can slow things down even when mostly empty

“Model Won't Load” Checklist

- Insufficient RAM/VRAM for the chosen quantization — try a smaller quantization level
- Corrupted or incomplete download — re-download the model file
- GPU backend mismatch — confirm you have the correct LM Studio runtime for your specific GPU vendor installed

“LM Studio Is Open But Continue Can't Reach It”

The App Being Open Isn't the Server Being On
This is the single most common connection failure. Confirm the local server toggle specifically is on (Developer/Local Server tab), not just that the LM Studio application window is open.

SECTION 09

Quick Reference Cheat Sheet

Task type	Recommended temp	Recommended quantization
Deterministic bug fix / QA sweep	0.0 – 0.2	Q4_K_M or higher
General coding / refactor	0.3 – 0.5	Q4_K_M
Brainstorm / creative / growth ideas	0.7 – 0.9	Q4_K_M is fine — quality of ideas matters more than precision here
Autocomplete while typing	0.1 – 0.3	Smallest model that's still fast (e.g. 0.5B)

Before Reporting a Local Model “Isn't Good Enough”

- Confirm quantization is Q4_K_M or higher, not something aggressively compressed
- Confirm context length is adequate for the task and not silently truncating
- Confirm temperature matches the task type (low for precision, high for creativity)
- Only after checking all three: it may genuinely be a task better suited to a paid model — see the Field Manual's routing table

---

Ghost Logics LLC ? Companion to the AI Dev Studio Field Manual.