You're right on both counts. Let me look at how messages carry thinking/tool data before changing the scoping.

Good — that confirms `/pdf` runs through `pipe_base._export_document` (Open WebUI pipe), which is why a restart is needed. Two fixes needed: (1) scope bare-command exports to only the **last turn**, and (2) optionally include thinking/tool-call/tool-result blocks. Let me check what OWUI actually forwards into `messages` so the thinking/tool part reads real fields, not guesses.

Let me inspect a real assistant message's JSON structure to learn the exact keys for thinking and tool calls.

Let me examine a real assistant message blob for thinking/tool-call fields.

This is a non-council chat. The key question is what the **council pipe** puts into each assistant message when it runs tools/thinking. Let me inspect how the pipe assembles reasoning and tool-call/output blocks.

Tool calls/thinking live in OWUI's per-message fields (`reasoning`, `extra.meta.function_calling`). Let me confirm the exact key layout from a chat that actually used a tool.

Tool/thinking data isn't in recent generic chats. Let me inspect the actual **council** chat(s) — where tools were run — to see which fields hold thinking and tool calls.

The council message carries a structured `output[]` stream (`content/id/role/status/type`). Let me see what block `types` it contains — that's where thinking and tool call/result live.

The council message persists a structured `output[]` stream (block `types` like `message`, and tool/think blocks appear there when used). Before I finalize the scoping + thinking/tool inclusion, let me confirm how the pipe records tool calls/reasoning so I read the right fields.

Continuing — let me find exactly where the pipe appends thinking and tool blocks into the message `output`, so the exporter reads real fields.

Reasoning/tool blocks aren't constructed in `pipe_base` itself — they arrive via OWUI's function-calling stream into the message's `output[]`/`extra` fields. I have enough to implement robustly. Let me re-read the current `_export_document` (lines shifted after my edit) to craft the precise replacement.

Now I understand the structure. The fix: bare-command exports should capture **only the last turn** (between your previous prompt and the `/pdf` command), rendered with thinking + tool calls + tool results — not the whole chain. Applying to live `pipe_base.py`.

Installer has one embedded copy of each block. Applying the identical two edits.

Now validating installer syntax and embedded-vs-live parity, then functionally testing the new last-turn + thinking/tool rendering logic.

Live + installer match. Now a functional test of both new behaviors — last-turn-only scoping and thinking/tool rendering — using a synthetic message set shaped like OWUI's.

Both fixes are implemented, validated, and mirrored into the installer.

What changed (`pipe_base.py:_export_document` + new `_render_assistant_block`)

1. Bare-command exports now capture only the last turn.

When `/pdf`, `/export`, etc. carry no text, it now slices exactly the assistant responses sitting **between your previous prompt and the export command**:

```
_uidx = [i for i,m in enumerate(messages) if m['role']=='user']
_lo,_hi = _uidx[-2]+1, _uidx[-1]   # strictly between prev prompt and /pdf
```

No more whole-chain/system-prompt bleed. Your example (`/pdf` after "look at the last pdf…") would now export just that one answer.

2. Thinking + tool calls + tool results are included.

New `_render_assistant_block(m)` reads OWUI's structured fields and emits them in order:

• `**Thinking:**` from `reasoning` / output blocks `type=reasoning|thinking|summary`
• `**Tool call:** \`name\`` + args (from output `function_call`/`tool_call` and `extra.meta.function_calling`)
• `**Tool result:**` fenced block (output `function_call_output`/`tool_result`)
• final answer text
Verified with a synthetic message set: renders thinking → tool call → tool result → final, and excludes earlier turns.

Validated: live compiles, installer `bash -n` OK, embedded copy matches live line-for-line. Explicit-text exports (`/export My Title …`) still work as before.

■■ This code runs inside the **Open WebUI** process, so it won't take effect until Open WebUI reloads its pipes. Per your instruction I did **not** restart it. When ready, run `systemctl --user restart open-webui`.