

PR #112 · Cron manager: enable/disable toggle and “Run now”

Issue #77 · branch feat/77-cron-controls · opened 12 Mar 2026 06:54 UTC by an autonomous worker · merged 07:37 UTC after review

AGENT DIFF 12 files, +1,182 / -209	REVIEW FIX 1 commit, 2 files	SEVERITY CAUGHT HIGH command injection	STATUS Merged after fix
---------------------------------------	---------------------------------	--	----------------------------

Verdict: merge after the fix. The agent ticked every item on its acceptance checklist, but the agent's rewrite of two API routes deleted an input guard that had been added in an earlier security fix, and built shell commands from a URL path segment. Restored and hardened before merge.

WHAT THE AGENT PRODUCED

- A toggle switch and a “Run now” button on each cron job card, with a confirmation dialog, toast feedback and optimistic UI with rollback.
- Rewrote `POST /api/cron/[id]/toggle` and `POST /api/cron/[id]/run`, which call the scheduler CLI.
- Added four UI primitives (switch, button, alert dialog, toast) and their dependencies.
- All eight acceptance criteria in the PR body were ticked by the agent.

WHAT THE REVIEW CAUGHT AND CHANGED

1. Shell command built from the URL, guard removed HIGH

The previous version of both routes validated the job ID against an allowlist (added on 16 Feb in security PR #56). The agent's rewrite dropped that check and interpolated the raw path segment into a shell string. Double quotes do not stop `$(...)` or backtick substitution, so a crafted job ID in the URL would run arbitrary commands as the dashboard's user.

Agent version (sanitized sketch)

```
const { id } = await params
// no validation
execSync(`scheduler cron run "${id}"`)
```

After review (sanitized sketch)

```
const SAFE_ID = /^[a-zA-Z0-9_-]+$/
if (!SAFE_ID.test(id)) return badRequest()
execFileSync('scheduler', ['cron', 'run', id])
```

Fix: restored the allowlist on both routes and replaced the shell strings with argument arrays, so the ID can never be parsed by a shell. The same change was applied to the toggle route, where the enabled flag from the request body had also been interpolated into the command.

RESIDUAL RISK

- LOW

 enabled is not type-checked as a boolean. It is now passed as a single argument, so it cannot inject, but a non-boolean reaches the CLI.
- LOW

 On failure the routes return the raw error message to the client. The dashboard sits behind an auth middleware, which limits exposure.
- LOW

 The rewrite dropped the calendar cache revalidation the old route performed, so the calendar view can show a stale job state until reload. Not restored in this PR.

LESSON FOR THE FAILURE MAP

An agent asked to *add a feature* to an existing file treated earlier security code as clutter. The ticked acceptance checklist said nothing about it. Every rewrite of a previously fixed file is diffed against the earlier security fix.

Sprint Security Report

Admin dashboard v2.0 (Next.js + SQLite) · 30 merged PRs, #95–#125 · 10–13 March 2026

30

PRs merged (1 duplicate attempt closed, #96)

3

shell-injection paths from request input, all fixed before merge

25

of 30 PRs got at least one fix or refactor commit before merge

2

residual shell-string sites left open, listed below

SUMMARY

Thirty issues were implemented by autonomous Claude Code workers, one branch and one PR per issue, opened in nine waves. The first PR opened on 10 Mar at 13:54 UTC and the last merged on 13 Mar at 09:35 UTC. Every PR was merged by hand after review. Three PRs passed user-controlled input into a shell; one of them did so by deleting a guard from an earlier security fix. All three were fixed before merge. Two lower-risk shell-string patterns on internal values were left in place and are listed as residual risk.

FINDINGS

PR	AREA	FINDING	SEV.	RESOLUTION
#112	Cron controls	Job ID from the URL interpolated into a shell command; earlier allowlist removed by the rewrite	HIGH	Allowlist restored, argument array
#111	Log search	Search query interpolated into a shell command. A character filter limited exploitability, but safety rested on one regex	MED	Argument array; “no match” exit handled
#121	Memory search	Same pattern as #111 in a second search endpoint	MED	Argument array; “no match” exit handled
#119	Project notes editor	File read/write path built from a route parameter with no containment check (path traversal)	HIGH	Name allowlist and resolved-path containment check
#117	GitHub stats	Repo string from the editable project registry passed to a shell	MED	Repo format validated; still runs through <code>sh -c</code> (residual)
#113	Disk usage	Config path interpolated into a shell command	LOW	Argument array
#102	Work queue	Drag-and-drop reorder sent a payload the API did not accept, so reordering silently failed	MED	Payload corrected, error state added
#95	Schema	Columns diverged from the spec; seed inserts not atomic; no FK cascade; duplicate active queue items possible	MED	Schema aligned, transaction, cascades, partial unique index
#109	Run diff viewer	PR number and repo from the database interpolated into a shell command. Not caught before merge	MED	Open (residual)

VULNERABILITY CLASSES CHECKED

Shell/command injection · path traversal · input validation at route boundaries · removed or weakened guards in rewritten files · data integrity (transactions, FKs, uniqueness) · silent functional failures. No test suite or CI pipeline existed in this repo, so every finding above came from reading the diff.

AGENT FAILURE MAP (EXCERPT)

What the agents consistently got wrong in this codebase, so the next sprint’s issue contracts and review checklist cover it up front.

PATTERN	SEEN IN	WHAT IT LOOKED LIKE	COUNTERMEASURE
Shell strings for CLI calls	#109, #111, #112, #113, #117, #121	Six PRs ran a CLI through a shell string, whether newly written or inherited from an existing helper	Ban shell strings in the project rules; grep every diff for exec
Deleting earlier security code	#112	A feature rewrite removed an allowlist added by a prior security fix	Diff rewritten files against their last security commit
Framework-version drift	#99, #109, #118, #119, #120	Route handlers written for the previous Next.js API (sync params, missing force-dynamic)	State the framework version and route conventions in the project rules
Client/API contract mismatch	#102	UI sent a payload shape the API it was calling did not accept	Review the UI and API sides of a contract together
Invented facts in docs	#125	Docs listed an API endpoint that does not exist and miscounted the route handlers	Check docs claims against the code before merge
Scope bleed	#95, #125	Build artifacts committed in #95; a docs PR also carried a type fix and a missing dependency	One concern per PR; artifacts in .gitignore
Duplicate attempt	#96 → #97	First attempt at issue #63 closed and redone	Counted, not hidden: 31 PRs opened, 30 merged

WHAT SHIPPED, WHAT IS LEFT

Shipped (30 PRs)

Work-queue schema, API and drag-and-drop UI; run history with drill-down, transcript and diff tabs; log viewer with filters and search; cron manager with controls, timeline and recent runs; disk-usage card; SSE live updates; projects registry and notes editor; memory browser and search; worker settings; PWA; DB retention; v2.0 docs.

Left open (residual risk)

- #109 diff viewer still builds shell strings from database values. Fix: argument array.
- #117 GitHub helper still runs through `sh -c` (kept for a `--jq` filter) behind a format check. Fix: argument array, filter in code.
- #112 calendar cache revalidation dropped; `enabled` not type-checked.

METHOD

Sample from my own admin dashboard, March 2026. Not client work. The dashboard monitors my own AI agents; the repository contains no client code or client data.

Each issue was dispatched to an autonomous Claude Code worker on its own branch, which opened one PR. Every PR was merged by hand after review. Findings were fixed as follow-up commits on the same branch, so the agent’s version and the reviewed version are both in the history. Every number and finding in this report traces to a specific PR and commit in that history.

Code excerpts are sanitized sketches; the CLI name and internal paths are generalized. Severity is my judgment for a self-hosted, auth-gated admin tool; a public-facing app would rate several of these higher.