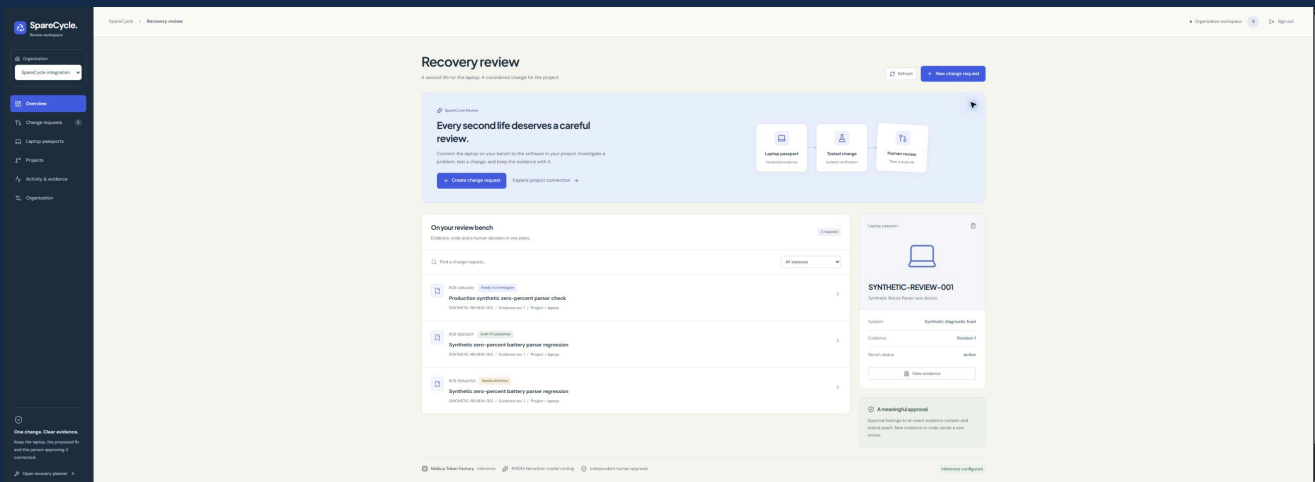


A second life. A reviewed change.

From laptop evidence to thoughtful repair plans and tested draft pull requests.



Nebius Token Factory supplies inference. NVIDIA Nemotron proposes and reviews. The application keeps tests, budgets and human approval in control.

A guide to the product

Read the working path first. Then assess the evidence and the investment gates.

The job worth solving	3
Workshop: make every part count	4
Pilot Operations: close the loop	5
Review: evidence to draft PR	6
NVIDIA Nemotron on Nebius	7
Tavily: useful public references	8
The safety is in the workflow	9
Real calls. Real tests. A draft.	10
Start a company pilot simply	11
A focused business opportunity	12
Invest in complete workflows	13
Try it. Inspect it. Judge the evidence.	14

The core proposition

A refurbishment coordinator needs a defensible allocation of scarce parts. A diagnostics engineer needs a reproducible defect translated into a narrow, tested change. SpareCycle brings those decisions into one product while preserving their different data and approval boundaries.

This dossier describes a bounded prototype and recorded synthetic pilot results. It does not claim certified hardware, enterprise certification, customer ROI, universal repository support, or hundreds of shipped features. Dates and proof links matter: a recorded run is not a fresh execution.

Browser-local Workshop

Reviewed inventory → constrained plan → technician workflow → exported evidence packet.

Authenticated Review

Organization evidence → bounded code proposal → unchanged tests → separate approval → draft PR.

The handoff between these paths is deliberate and human-reviewed. There is no implied shared identity, inventory synchronization or automatic physical release.

The job worth solving

Help teams move from an uncertain report to the next justified action.

At the repair bench

A batch of laptops arrives with uneven evidence, limited parts and limited technician time. A seemingly easy repair can consume the component that would complete several other devices.

The coordinator needs a complete allocation, visible trade-offs and clear reasons to hold a device. A plausible AI answer alone is insufficient.

In the diagnostics project

An engineer sees a fault report that may come from hardware, missing evidence or a software parser defect. A broad code agent may change more than this case warrants.

The engineer needs attributable evidence, unchanged tests, a bounded candidate and a second person who can approve the exact change.

One product, two accountable workflows

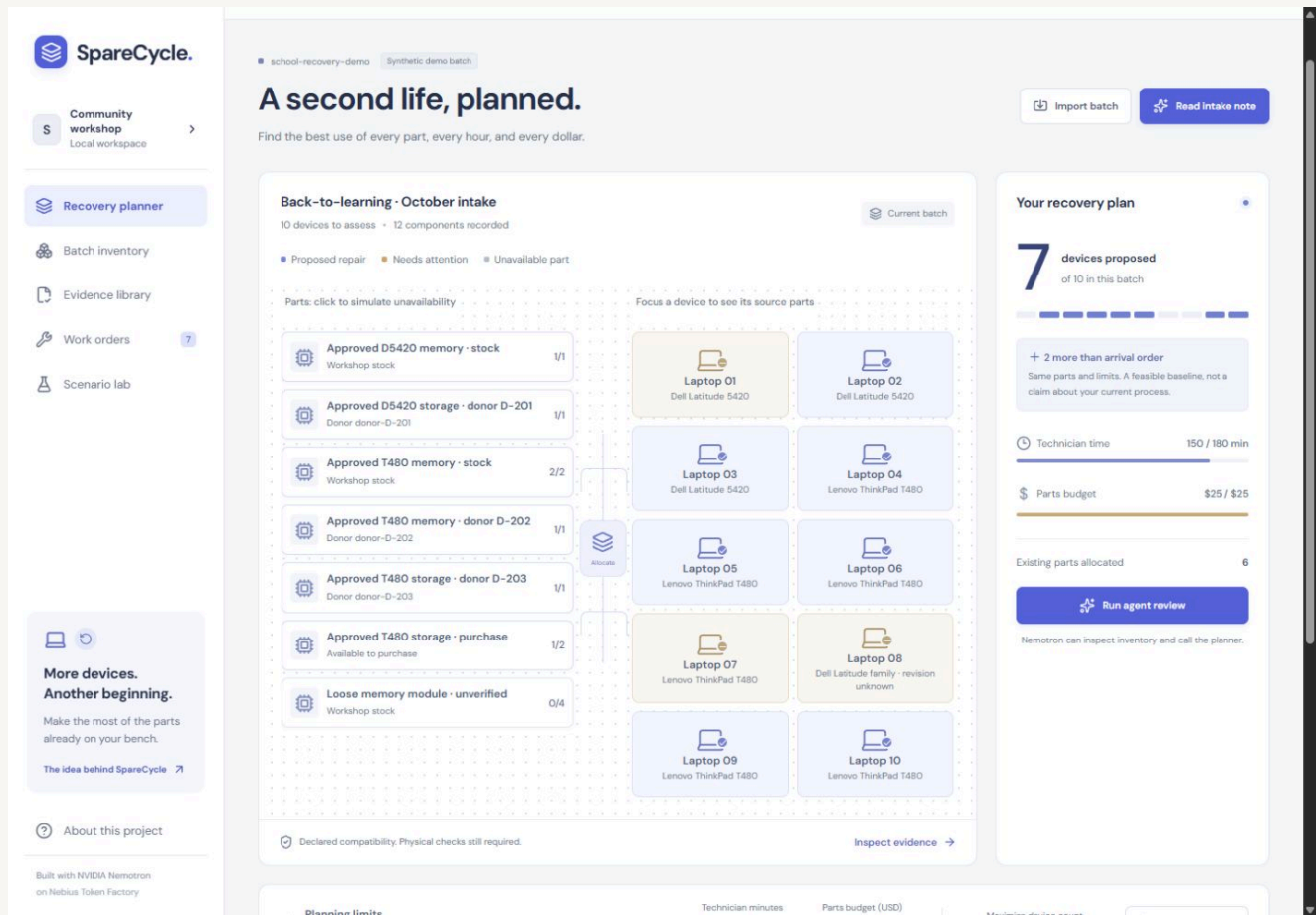
Workshop proposes how to use declared inventory. Review turns a supported diagnostic defect into a draft PR. The pilot joins them through a clear human workflow, not through an invisible automatic synchronization.

- 1 Capture what is known**
Record the model, issue and actual evidence. Preserve unknowns rather than manufacturing diagnoses.
- 2 Propose an action within bounds**
Allocate declared parts or request a narrowly scoped code proposal. Respect holds, budgets and supported profiles.
- 3 Require evidence before progression**
Physical work needs technician checks. Code work needs a failing baseline, passing candidate and separate approval.

First customers to validate: community refurbishers, school IT teams and organizations maintaining endpoint diagnostics code. These are buyer hypotheses, not existing customers.

Workshop: make every part count

A visible plan for an explicitly bounded batch.



Genuine application capture with a synthetic demonstration batch. The seven proposed devices and two-device comparison shown here belong to that fixture, not a measured customer improvement.

What it does

Validate supplied inventory, exact declared part identifiers, evidence links, holds, stock quantities, purchase spend and technician time. Compare with a feasible arrival-order baseline and replay selected shortages.

Nemotron can read intake notes and call the supplied read-only inventory/planning tools. The deterministic planner owns allocation and totals. An exhausted search limit is shown honestly.

What stays with the team

Confirm physical condition and compatibility, perform repairs, sanitize storage and complete QA. Local company profiles stay in this browser; shared warehouse transactions are outside the Workshop prototype.

Pilot Operations: close the loop

Turn a frozen proposal into a reviewable record of what the team actually did.

Release status

Implemented and browser-checked: frozen plans, assignment, work records, blockers, referenced manual checks, scorecards and recoverable local exports. Full current checks gate declared checklist readiness; changed work and recovered copies reset that readiness.

Freeze the plan

Preserve the input, constraints, allocations and timestamp. Later batch edits cannot silently alter an earlier pilot.

Run the bench

Assign a technician and due date. Record queued, active, blocked and reported completed work with reasons and history.

Record evidence

Separate effort and incremental costs from estimates. Record manual function, sanitization and accessory checks.

Explain the outcome

Compare proposals with reported completions and missing outcomes. Export a handoff packet with its limitations.

A bridge for people

The coordinator can hand an evidence packet to a diagnostics engineer when a supported software issue appears. Creating an authenticated Review case remains an explicit action. A local pilot record never grants access to an organization or a repository.

Useful status, honest meaning

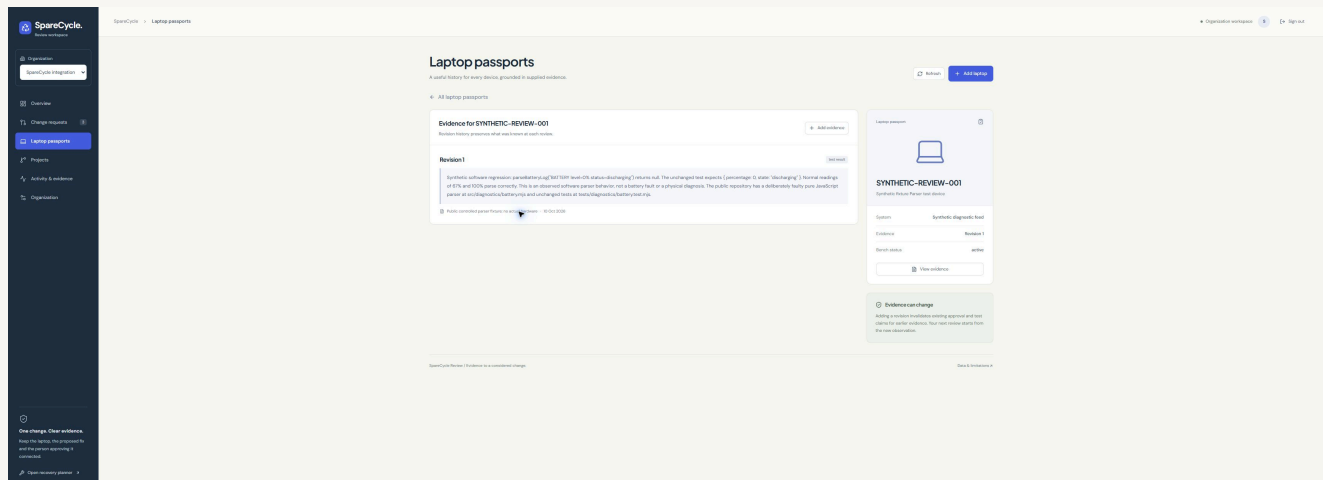
"Completed" means the technician recorded work. "Checklist ready" requires current functional, sanitization and accessory checks with evidence references. Changed work invalidates old checks. Neither status is independent hardware certification.

The important boundary

Pilot Operations is browser-local. It does not consume shared inventory, perform an erase, send messages, issue a compliance certificate or create a protected release approval.

Review: evidence to draft PR

Bind the laptop record, code snapshot, test result and human decision.



Actual Review interface. Passport fields and diagnostic evidence in the recorded pilot are synthetic. A passport is a supplied record, not telemetry from an installed laptop agent.

- 1 **Record the case**
An organization member creates a laptop passport, appends evidence and opens a change request.
- 2 **Connect the permitted project**
The GitHub App and connecting person must both have the required repository access. The selected source and tests are pinned to a commit.
- 3 **Investigate and propose**
Nebius Nemotron classifies software, hardware or insufficient evidence. Only an eligible software case proceeds to a scoped parser patch.
- 4 **Test, review and approve**
Run the unchanged tests, obtain a fresh model review and require another authorized person to approve the exact candidate.
- 5 **Create a draft**
The trusted service creates a draft PR through the GitHub App. It does not merge or deploy the change.

Review insights: filter loaded records, compare evidence and proof references, and preview a redacted JSON or spreadsheet-safe activity handoff. The queue does not rerun tests or grant permission; existing server checks remain authoritative.

NVIDIA Nemotron on Nebius

Stage-based routing gives each model a clear job and a bounded request.

Tier	Job	Recorded scope
Nano	Selectable investigation	Used in the recorded private case and a separate production investigation.
Lightning	Default investigation and Workshop assistance	Recorded public investigation and read-only planning workflow.
Super	Patch writer and fresh reviewer	Non-Ultra cases route these stages to Super; both completed in the private pilot.
Ultra	Explicit escalation	Operator opt-in; bounded investigation test recorded. No quality superiority benchmark.



The harness keeps the authority

The model returns structured proposals. It receives no shell, network, file-write or publishing tools in Review. The trusted service validates outputs, runs the allowed operations and records the result.

Maximum Review JSON context is 100,000 bytes. Provider calls have a 40-second deadline and no automatic retries. Patch output is bounded to 8,192 tokens; investigation and review to 4,096. These are ceilings, not measured consumption for every run.

Exact catalog identifiers and observed roles: [backend evidence](#). Omni, vision and dedicated models are not integrated merely because a provider catalog lists them. This is not a simultaneous four-model debate.

Tavily: useful public references

Research official documentation, then ask Nemotron to explain only the supplied evidence.

Runtime verification status

Production runtime verified on 11 October 2026 at 05:21 UTC: HTTP 200, five model-matched T480 sources and one reported Tavily credit. NVIDIA Nemotron Lightning on Nebius returned the referenced summary; reported latency was 2.324 seconds.

Recorded Tavily request: cbbb55f0-2a43-439d-b944-b5c33aa33781. Explicit keyless search mode; full manual scope remains unverified. [Production release ledger](#).

- 1

Ask a public model question
A technician enters a model name and a documentation question. No passport, serial number, inventory or project code is automatically included.
- 2

Call Tavily at runtime
The server performs one bounded search, restricted to configured official support/documentation domains. Up to five accepted results become source IDs.
- 3

Synthesize on Nebius
Nemotron receives plain-text snippets as untrusted data. A strict result schema requires findings to reference existing source IDs.
- 4

Review the original source
The UI shows source links, snippets, uncertainties and next checks. A technician confirms applicability to the actual device.

Control	Effect
One search; 10-second search budget	No silent retry loop or open-ended browsing.
Five results; 128 KiB response ceiling	Bounded provider material before model synthesis.
Official-domain and URL checks	Unsafe links and unsupported domains are discarded.
Server-assigned citations	The model cannot invent a clickable source URL.
Empty evidence means abstention	No uncited fallback presented as research.

Citation-ID validation proves a reference exists, not that every generated sentence correctly interprets it. The source text and physical applicability still need review. [Research contract](#).

The safety is in the workflow

Keep identity, inference, execution and publication in distinct boundaries.

Identity and records

Supabase validates the user. Organization membership and role-aware database functions protect records. Public configuration is not a health claim.

Inference

Nebius receives bounded case material. Strict schemas reject incomplete output. LangSmith is configured for operational metadata with raw inputs and outputs hidden.

Execution

GitHub Actions runs original and candidate in separate restricted, digest-pinned containers. Credentials stay outside the test containers.

Publication

A different member approves the bound evidence version, base commit and patch hash. The GitHub App creates only a draft PR.

Controls that matter together

- Tenant membership and repository access are checked separately.
- Durable Review reservations and job leases bound concurrent work and estimated spend.
- Provider uncertainty is surfaced rather than silently retried.
- Source policy restricts the candidate to a pure diagnostic parser.
- Test proof binds source, candidate, evidence, named tests and runner identity.
- Changed evidence or code invalidates the path to an old approval.

What these controls do not establish

They do not certify enterprise security, guarantee absence of prompt injection, prove all program behavior or establish physical repair. The Nebius Sandboxes adapter exists in source but the demonstrated execution provider is GitHub Actions.

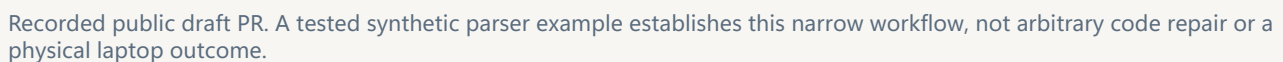
The recorded pilot uses synthetic inputs with genuine provider and GitHub execution.

3 / 4

The existing suite demonstrates the reported defect.

4 / 4

The same tests run against the candidate. Their contents stay unchanged.



Public draft PR 1 and public verifier run are accessible evidence. The private pilot used a pinned verifier in the same private repository; its proof completed on 11 October 2026 at 03:29 UTC.

The detailed [evidence ledger](#) contains timestamps, digests, inspected trace scope and limitations. A screenshot or replay is not a fresh execution.

10 / 14

Start a company pilot simply

Use the public Workshop first; provision the protected code workflow deliberately.

Workshop in a few steps

- 1 Open the demo**
Visit the Workshop and inspect the clearly labeled synthetic batch.
- 2 Create a local workspace**
Name the company and set planning limits. Import supported CSV or JSON.
- 3 Review and export**
Check declarations, run a proposal, explore shortages and retain a local backup.

[Open Workshop](#)

Review with a second person

- 1 Sign in and organize**
Use enabled Google sign-in; create an organization and invite a reviewer.
- 2 Connect the selected repo**
Install the GitHub App for the intended repository and prove repository access.
- 3 Run the bounded case**
Add evidence, investigate, patch, verify, review and independently approve a draft.

[Open Review](#)

Private repositories require preparation

An operator provisions a trusted workflow, pinned commit, transport key and explicit binding in the selected private repository. The supported API accepts one pure JavaScript diagnostics module and one to four unchanged test files. This is not arbitrary-project self-service execution.

Google authentication has a recorded live return test. Customer SMTP is deferred; optional custom OIDC is not enforced company SSO. Corporate SSO, SCIM, retention policy, restore drills and support ownership remain rollout gates. [Private setup guide](#).

A focused business opportunity

Sell a measurable operational result after demonstrating it with a real team.

Potential buyer	Job to validate	Pilot measure
Refurbishment coordinator	Allocate scarce stock and explain deferrals	Input preparation time; completed outcomes; plan changes; evidence gaps.
School or nonprofit IT lead	Prepare reliable devices for reuse	Time held for missing evidence; completion and aftercare records.
Endpoint diagnostics team	Turn a reproducible parser defect into a reviewable fix	Accepted evidence-to-draft time; human review effort; regressions; abstentions.
Managed service provider	Repeat a constrained workflow across clients	Onboarding effort; tenant isolation tests; per-accepted-case cost.

Commercial hypotheses

A per-team subscription could cover the workspace, review history and operational reporting. A transparent usage allowance could cover research and model stages. A separate implementation service could cover qualified private verifiers and data migration. These are pricing hypotheses; no revenue or willingness-to-pay result is claimed.

The unit economics to measure

Count search credits, model usage, verification runtime, storage, support effort and human review. Compare cost per accepted outcome with the actual existing process. Rejected and abstained cases consume resources too.

The investment test

Would one company use this workflow repeatedly, provide representative evidence, and pay after including setup and review time? Demonstrate that before expanding to fleet administration or general code review.

Invest in complete workflows

A 200-capability execution map, with dependencies and acceptance gates.

Map	Module	Dependency / investment gate
EX01	Evidence lifecycle	Provenance, artifact storage, technician policy
EX02	Intake and asset control	Shared identity and receipt ledger
EX03	Parts and compatibility	Vendor authority and qualified review
EX04	Planning and resilience	Validated constraints and frozen inputs
EX05	Bench execution	Atomic stock and authenticated work history
EX06	QA and physical release	Real test evidence and independent sign-off
EX07	Procurement and economics	Supplier evidence and financial approvals
EX08	Company administration	SSO lifecycle, recovery and retention
EX09	External integrations	Scoped credentials, contracts and replay
EX10	AI evaluation and operations	Held-out fixtures and measured reliability

Each module contains 20 selected, meaningful planned capabilities with a stable ID and observable acceptance criterion. These are curated from the existing backlog, not 200 additional shipped features. [Read all 200 acceptance gates.](#)

The next investment sequence

First: observe a real workshop and diagnostics team using the supported workflow; measure the local Pilot Operations handoff and accepted code outcomes.

Then: qualified shared records, email delivery, company identity lifecycle, restore drills and independent security assessment.

Expand only with evidence: a second language profile, one read-only device-management connector, measured routing evaluations and controlled fleet rollout. Physical diagnostics and environmental claims need external ground truth.

Try it. Inspect it. Judge the evidence.

The product is useful when each next action has a reason and a record.

Live application

sparecycle-topaz.vercel.app

Workshop planning, authenticated Review and the documented demonstration.

Product and source

[Workshop](#)

[Review](#)

[Video demonstrations](#)

[Source and setup](#)

[Judge access and test paths](#)

Evidence and next work

[Backend evidence ledger](#)

[Tavily contract and limitations](#)

[Dated release verification](#)

[200-capability delivery map](#)

[Investment priorities](#)

The claim to remember

SpareCycle demonstrates a bounded chain from laptop evidence through Nebius Nemotron, isolated unchanged-test verification and independent approval to a draft PR. The Workshop complements it with inspectable repair allocation and explicit uncertainty.

The next product milestone is a repeatable customer workflow with measured outcomes. The roadmap is larger than the current build, and the dossier keeps that difference visible.