Warranty *Vault*

UGREEN Hackathon 2026 · Round 1 Project Proposal · Skill name: warranty-vault · Demo: warranty-vault-sama-5806.vercel.app · Source: github.com/demir-sama/warranty-vault

01 What does your Skill do?

Warranty Vault reads the receipts, order emails, manuals and serial-label photos already on a UGREEN NAS and builds a household asset register with return-window and warranty alerts, citing a source file for every date.

02 Who is your target user, and what is their problem?

Households buying electronics, appliances and furniture. The paperwork ends up scattered across Downloads, mail exports and phone photos. When something breaks, people can't find the receipt, serial or warranty term in time, and return deadlines pass unnoticed.

03 What is your solution?

One request ("What's still under warranty?") produces a register: item, serial, retailer, price, purchase date, last return day, warranty end, and the files that prove each. Urgent items come first. Saving the register and tagging source files each require approval.

04 What is the main user flow?

User names folders → file-agent finds documents and checksums → duplicates set aside → photo-agent OCRs photos → document-agent classifies, extracts and reconciles → Skill ranks alerts → user approves saving the Markdown register (knowledge-agent) and tagging files (file-agent).

05 How is your idea different from existing tools or common AI solutions?

Receipt apps need you to re-scan purchases from now on; Warranty Vault starts from the history you already have. Chatbots guess warranty terms; here every field carries its source line, unwritten terms are labelled "assumed" with the regional rule used, and receipt-versus-label serial conflicts are flagged, never resolved silently. Nothing leaves the NAS.

06 Which UGREEN AI NAS features does your Skill use, and at which step?

ug-filesystem find/md5sum (step 1–2), ug-photo vision-ocr (3), ug-document metadata (4), extract (5), reconcile (6), ug-knowledge save-text (8, approved), ug-filemgr tag-get-by-name/tag-add/tag-add-file (9, approved).

07 What is your technical approach?

Markdown-only orchestration Skill delegating to four registered experts. Inputs: folders to search, optional save folder. Permissions: read within named folders; writes only after explicit approval; no delete, move or share. Extraction schema requires a verbatim source line per field and nulls for unwritten terms. Warranty priority: receipt, manual, then a labelled regional default. All five delegation states are handled. Models: the NAS experts' built-in models; no outside AI or network calls.

08 What have you built so far? What is not built yet? What still needs testing on a NAS?

Built: SKILL.md and six references, passing uliya-skill-kit 0.7.3 static check with 0 errors and 0 warnings; a public demo (warranty-vault-sama-5806.vercel.app) running the same workflow on a simulated NAS with 12 items from 8 countries. Not built: runtime on a real NAS. Pending NAS validation: expert output formats, OCR on real receipts, tag creation.

09 How will you test your Skill if you become a Finalist?

Core test 1: mixed folders with receipts, manuals, duplicates and non-receipts produce the correct register and skip list. Core test 2: approval gates: declining save and tag writes nothing; approving writes one new file and tags exactly the source files. Edge case: receipt and label serials disagree; the asset is marked "Needs you" with both files.

10 What are the known limits, risks, or outside dependencies of your Skill?

Accuracy depends on NAS OCR and document extraction quality, especially faded thermal receipts. Regional defaults are not legal advice. No scheduled reminders, since the catalog has no scheduler; users re-run the Skill or import the calendar export. Depends on experts behaving as the published catalog describes.

Third-party components and licences (excluded from word count): The Skill package is Markdown only and bundles no third-party code or data; it relies on UGREEN's registered NAS experts. The separate web demo uses Next.js, React, Tailwind CSS, Framer Motion and Zod (MIT) and the Vercel AI SDK (Apache-2.0). An AI coding assistant helped draft and review the Skill and demo; the Skill was checked with the official uliya-skill-kit 0.7.3. Answers total under 500 words.