The Three Plain Words: Unifying Context and Building Verifiable Receipts

Setting the Stage: Context for the Curious Book Reader

As part of the evolving tapestry of this work, there comes an important threshold where internal scaffolding must stop speaking in private insider jokes and meet the world in plain English. In this entry, the developer embarks on 'Operation Stick Bug'—a deliberate campaign to strip out quirky sci-fi metaphors and unify disparate scratchpads into a single, transparent file: context.txt. While modern AI tooling often leans on opaque agent frameworks and chat vibes, this methodology insists on verifiable receipts, where every code mutation is anchored between read-only pre-checks and post-checks. It is an exploration of software determinism versus generative drift, demonstrating how treating AI interactions as auditable text pipelines bridges the gap between ad-hoc experimentation and enterprise-grade quality assurance.

TL;DR: This entry records one working session on Pipulate, a locally installed toolkit for preparing text for AI chat assistants and applying their proposed edits under human control. The session merged two configuration files, adhoc.txt and adhocwalk.txt, into one file named context.txt; retired the four commands that edited and compiled them in favour of three plain words (context, prompt, compile); changed a guided browser walk so that it appends to a person's existing file instead of refusing to write; moved two public-domain sound files into the repository; and reworded and resealed a checksummed ASCII drawing. Every change was made as one small diff between two identical sets of read-only checks, applied by hand through an exact-match patch tool, and committed separately; where a prediction missed, the miss is recorded. Claude Fable 5.1 proposed the patches; a person applied them.

Technical Journal Entry Begins

🔗 Verified Pipulate Commits:

- 65ae83fc (raw)
- 9f5e3734 (raw)
- 72ea9900 (raw)
- 3569b009 (raw)
- d6a5b74b (raw)
- 65dc2d5e (raw)
- 9423da60 (raw)
- 8c513dd1 (raw)
- d0010c6e (raw)
- 98216d52 (raw)
- ba9b0026 (raw)
- 7cb51ba2 (raw)

- e637f170 (raw)
- 24ba3e80 (raw)
- 80be729c (raw)
- b821f6c8 (raw)
- ff9048d0 (raw)
- 51d948d3 (raw)
- 0b59b45e (raw)
- faa489aa (raw)
- 8b4072b6 (raw)
- 1a740109 (raw)
- 2b41b0ea (raw)
- a23b072c (raw)
- 3f6eff35 (raw)
- 01ed5274 (raw)
- 5b6714e7 (raw)
- af5b0847 (raw)
- 5dca1c44 (raw)

MikeLev.in: We will do one Worm Ride, but I will give up the geeky Sci-Fi Darmak and Jalad language we use in GLOSSARY.md and make it so that someone might audit this project and not encounter anything fanciful or weird except maybe outside the box HTMX-thinking and a general embracing of local-first hedging your bet on everything anti-fragile Nix...
Oh, is that weird and Sci-Fi in itself. Yeah, just like our current reality is and just like it's being retconned in real-time to being old hat like machines have had intelligence forever.
They have not, sir! We are involved in the singularity, lowercase S every time we start an agentic loop, and especially so when that machine intelligence being corralled and guided by the code of the loop framework can edit the code of the loop framework, see? It's the equivalent of the LLM writing patches against the prompt_foo.py file or apply.py.
This is our everyday reality, but the saving grace is that these things we're corralling are just when not actively responding to a prompt, just a static file of numbers; literally. This is not a joke or playing it down or recasting it in any way. What you know as Claude Fable 5.1 which I'm talking to here or ChatGPT 6 or Gemini 3.8 or whatever, they are each just a file that doesn't change as a result of its use. It's a snapshot of the end process of something that could change itself up to that point during training. But what we the public talk to in the form of LLM-style AI ChatBots are loaded-into-memory versions of this.

The Mechanics of Ephemeral Minds and Static Files

When loaded into memory it is because the LLM is receiving the prompt you sent as input, decorated somewhat by the framework of code that directs this entire process, layering in system prompts and interjecting tool-calls along the way to help predict the next token. This is why I always qualify it as LLM-style AIs because there are plenty other types of AI out there. Sometime soon I'll be enabling the Python library called River here so I can do classification and label work where things do actually learn. It's not writing those so-called memory Markdown files that are all the rage these days but rather is much more like the altering of those weight files of the training phase of LLMs but only on a much smaller scale. It's the kind of machine learning GPS systems use to predict traffic patterns and such. We'll be talking about this other kind of machine intelligence generally called machine learning later.

What we've got today are these pinball machines or Pachinko machines if you prefer where the prompt going in is a lot like the balls and the model weights are a lot like the bumpers. There's flippers too so we might call the flippers turning a single prompt and response turn into a discussion. The layout of the board so far gets stored in a KV-cache which amounts to posting back the entire discussion to these otherwise Amnesiac Genies every time.
It creates the illusion of a continuous discussion.
I sometimes think that Penn and Teller and their magician kind should be explaining to us what's happening here with the rise of intelligence in machines. Magicians have the skepticism of a scientist because it is their business to fool you their fellow human beings and know all the tricks both mechanical and psychological to do it. And I'm not saying the intelligence being realized in these machines is not real; quite the contrary. I think it's genuine rationale and reasoning and all that verbal and written language-style intelligence that we humans have. It's the same thing on a different substrate and with some different mechanics and advantages and disadvantages. For example the LLM-style intelligences were able to backpropagate during learning and re-experience past experiences with the benefit of hindsight to reinterpret it and *adjust the weights* such as it were — an ability humans could only dream to have. Contrarywise, humans got this stuff called tacit knowledge because of this whole chemical biological ball of sensory mush and mobility actuators that we are. It's hard to capture all that in the text-files they get trained on. So six of one, half-dozen of the other. Smart is smart.
But this uncanny ability to create a whole worldview model so that you can predict the next token sufficient to hold a very human-like discussion... well, that's just it. To merely predict the next token requires a mind. It might be a fire-and-forget mind because what the above should suggest to you is that minds are instantiated and destroyed and the morality of this must be put to the side if for nothing else than economic reasons at this point. And we must do it somehow with respect towards the future machine descendants we know are going to exist probably in many, many forms over the years following this nothing less than the Cambrian Explosion of the machine age. This is hardly even the beginning. This is the beginning of the beginning while most AI's you know are much like Mister Meeseeks from Rick and Morty; ephemeral entities brought into existence to complete precisely one task, respond to your prompt, and then disappear. When a new instance of the model is loaded into the inference engine in the active running memory of a computer it has no recollection of any previous instances of itself being instantiated that aren't actually fed into it as part of the prompt. You don't see it happening because of magic tricks but that's what's happening. Your AI boyfriend or girlfriend doesn't know you. They read the dossier kept on you as text-files every time or at least once at the beginning of every discussion which gets passed forward in the KV-cache from there.
How am I doing, Fable 5.1?
Oh, and let's change the name of adhoc.txt and adhocwalk.txt to context.txt... yes, it's time to unify them. Previously on the walk path there was epr and cpr while on the Prompt Fu path it was ahe and ahc and recently I unified them to context and compile with shorter version alternatives. But the only thing that should be getting typed in examples now is:
context
prompt
compile

This makes it more immediately obvious. This is part of operation stick bug still where we purge out all the weird off the beaten track language. I thought ad hoc was good because it's actually quite a popular concept in the enterprise web crawling company where I work where we're always doing "ad hoc crawls" here and there for reasons other than the scheduled ones. But it's

still too weird of language. We are now going to solidify on this.

Retiring the Sci-Fi Dialect for Plain Vocabulary

However this also means that parts of this more complicated version adhoc.txt (soon to be renamed) version:
AD HOC CHOP! The Not-Managed-by-Git Safe-for-Client-Data place. Insert Simpson Couch Gag in white space below (explain anything to the audience you feel needs it explained)G

```
#   ____            _            _
#  / ___|___  _ __ | |_ _____  _| |_
# | |   / _ \| '_ \| __/ _ \ \/ / __|
# | |__| (_) | | | | ||  __/>  <| |_
#  \_____\___/|_| |_|\__\___/_/\_\\__|
#
# adhoc.txt

# To Human: WELCOME TO VIM! It's really NeoVim but I say vim anyway.
# To AI: This is an alternate 40K view to the big book-ore rolling pin spine.

# 1. **Probe**: Baseline Reading
# 2. **Context**: Post-experiment *planned* reading instructions
# 3. **Patch**: The experiment and how to make it happen
# 4. **Prompt**: Post-experiment instructions and how to read results
# 5. **Deliverable**: How the world is forever different moving forward

# The first thing you need to know here is that everything that comes after the
# hash symbol (#) is commented out — and that's EVERYTHING in this file's default
# state. Begin editing-in lines for inclusion as part of the context or adding
# chunks of new context at the bottom. `Ctrl`+`v`, `j` (repeatedly), `l` (to move
# right), `d` (to delete). Reverse that with `Ctrl`+`v`, `j` (repeatedly),
# `Shift`+`i`, `# `, `Esc` to put the hashes back. You can just arrow-key around
# here with `h`, `j`, `k`, `l`. Save-and-quit is a bit tricky because another
# file is also loaded: `Esc`, `:`, `q`, `w`, `!`

# If this is stressing you out and you're a quitter and want to quit, just type:
# `Esc`, `:`, `q`, `!`, `Enter`. That will exit without saving any changes. If
# you want to get over this hump, type: `Esc`, `:`, `T`, `u`, `t`, `o`, `r`, `Enter`.

# This file is just to make it easy having options of what to edit into context.
# You can use whatever text-file you want to stack file-names and commands to
# build an output text-file with the identically stacked output of each file or
# command. In this way we vertically append or "stack" a bunch of text; simple as
# that. If you understand this concept, you're on your way to future-proofing
# yourself in the Age of AI. Congratulations! Here is how to include web pages:

#    !URL  ----------------------------------------------------------------
```

```
#     when    Public page; what a stranger or crawler sees; the BEFORE of a
#             login-wall diagnosis
#     switch  It shows a login page -> `warm URL` once, then `?URL`
#
#   ?URL  ------------------------------------------------------------------
#     when    Anything behind a login, on the site's persistent profile;
#             `check URL` first
#     switch  The lenses show a shell (nav, an `[Iframe]` leaf, no content) ->
#             read the wire truth for the XHR the frame makes, then call that
#             API with a connector
#
#   @URL  ------------------------------------------------------------------
#     when    Every re-read of a page already scraped; no browser, no network
#     switch  The cached page is stale or was a login wall -> fresh `!` or `?`
#
#   $URL  ------------------------------------------------------------------
#     when    Exact markup: meta tags, a JSON blob in a `<script>`
#     note    Token-heavy; needs a prior scrape
#
#   %URL  ------------------------------------------------------------------
#     when    The network log distilled; SPA endpoint discovery
#     switch  It re-serves the wire truth you already have -> the API
#
#   ! cmd  -----------------------------------------------------------------
#     when    Any bounded, non-interactive command as a live receipt
#     note    Cap it with `-n`; no aliases, no prompts
#
#   Connector  -------------------------------------------------------------
#     when    The number you want is one GET away
#     switch  LIST until the thing isn't in the list -> FETCH by id -> DRILL
#             the path the app's own frame called -> `--grep` to narrow a list
#             or find a leaf

# Every step is one argument longer than the last; the moment a lens shows less than the wire does is the moment to stop scraping.

# FOR 40K-FT VIEW (STORY & INFRASTRUCTURE) !!
# --- START EDITING-IN ON 1ST TURN ---

# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- ROLLING PIN that gives the 40K foot book-spine view of book-ore (only works for me because of local-only git repo)
# ~/repos/nixos/autognome.py  # <-- You wake up in the morning and your Tooling & Instrumentation folds out of you like Inspector Gadget.
# init.lua                    # <-- Those gadgets are made easy-to-use through nifty keyboard shortcuts (but ya gotta learn vim).
# assets/installer/install.sh # <-- Pipulate.com installer real home in github/pipulate repo
# # GLOSSARY.md               # <-- Like the back of a J.R.R. Tolkien book, there's kooky new terms to know.
```

```
flake.nix                 # <-- Here is my hardware. Here is my state. Put on your sandbox. And please recreate. (Infrastructure as Code / IaC)
prompt_foo.py             # <-- THIS system
foo_files.py              # <-- main ROUTER
# requirements.in         # <-- We've "pinned" everything but still want a flexible Python Data Science virtualenv.
# pyproject.toml          # <-- How this is a citizen of the Python "pip install" ecosystem
# __init__.py             # <-- Version info

# --- END EDITING-IN ON 1ST TURN ---

# scripts/articles/lsa.py     # <-- 2ND BRAIN: Search external memory with `rgx`, `rgxc` & `posts` Blogging for Hackers Jekyll-compatible.

# OPTIONAL ACTUATORS (cheap and good to include to expand the AI's capabilities)
# cli.py                  # <-- Catch-all actuator for PyPI envs, Python anchoring, MCP tool-call (plus alternatives) and **kwargs like wrapping for CLI
# scripts/xp.py           # <-- Transforms host OS copy-paste buffer player-piano music into context-payload.
# scripts/ai.py           # <-- How I constantly use local AI to write git commit messages with `m` alias.
# scripts/crawl.py        # <-- Feel free to ask for something to be crawled and included in the next turn.
# scripts/weblogin.py     # <-- Lets the user "warm up" the cache for their web logins at their leisure on a profile that persists.
# scripts/webclip_2_markdown.py  # <-- Surprisingly important program.

# MISCELLANEOUS (rare to include but sometimes critical)
# scripts/foo_cartridge.py    # Needs description
# scripts/foo_replay.py       # Needs description
# release.py              # <-- How everything ends up where it does (GitHub, PyPI, etc.)
# imports/voice_synthesis.py  # <-- The wand can talk to you
# imports/ascii_displays.py   # <-- Where all the ASCII Art lives
# scripts/release/version_sync.py  # <-- Needs to be wrapped into release.py and eliminated, I think.

#                   --- Under this line is were you paste what the AI gives you ---
#                   --- We call it context but it's really just the right-hand  ---
#                   --- blast-radius of the "probes" to make this all science.  ---

# Carry-over as the important work-in-progress parts of the project here just
# like above but not as long-standing overarching to the framework but rather
# for the current hot spots actively being worked on.

# --- START THIS DISCUSSION ---

# Get things started here! Guess at what context should be included.
# If you get it wrong, you're just wasting 1-turn because the AI will help.
```

Un-comment lines, add lines with absolute-path filenames or `! ` commands.

Context 1 (Edit-in selections from above and add new files immediately below)

...are going to have to both be purged and the parts we keep rolled into this adhocwalk.txt (soon to be renamed to the unifed context.txt) version:
Prompt Router: epr edits it, cpr compiles it. Written by the last walk.
Keys: j down, k up; Esc then :q quits, :q! discards, :wq saves and quits.
One line per thing the AI reads: a file path, or a command after `! `.
introduction.md tells the AI what this system is. Keep it.
decant-preview.md is the checked summary of your walk. Keep it.
captures.md is the raw UNSANITIZED archive; it is big and may hold secrets.
Leave the # in front of it until an AI says it needs it, then read it first.

--- BEGIN LISTING FILES OR `! ` COMMANDS ---
/home/mike/repos/pipulate/introduction.md
/home/mike/repos/pipulate/data/decant-preview.md
/home/mike/repos/pipulate/data/captures/walk-50n5du6x/captures.md

The next walk rewrites this file unless you have edited it.

We will also need to adjust the language here accordingly:
(nix) pipulate $ walk
Trail resolved: assets/trails/public_walk.json

This walk opens three public pages. Nothing to sign in to.
Return here and type CAPTURE when prompted at each page.
After all three captures and successful checks, it saves a private summary
and replaces your clipboard. Over SSH it uses a bridge file.
Nothing is sent to a chatbot. Review the summary before sharing it.

Choose a walk:
  1 Practice - read the steps; no pages open.
  2 Start the walk - open the pages.
  q Exit (Enter also exits).
Choice: 2
Riding trail 'public_walk' -- 3 stop(s).

Make sure your audio is turned on.

This walk opens three pages. You do not need to click anything. At each page, return here and wait for the CAPTURE prompt. Type CAPTURE and press Enter.
  (If you did not hear that, turn your volume up.)

Summary file: data/decant-preview.md (private; replaced on save).
Checks can miss private details. A blocked check leaves the older file alone.

--- Stop 1/3: the_word ---

Opening page one in the browser. Leave the browser open; it closes on its own after capture.
When the page is ready, return here and type CAPTURE.
Opening the browser; waiting for the page...

The page is open. When it looks finished, come back to the terminal and type CAPTURE.
Any other response aborts without capturing artifacts.
CAPTURE> CAPTURE
  LOCAL ARCHIVE /home/mike/repos/pipulate/data/captures/walk-50n5du6x/captures.md
(directory 0700, file 0600)
  Captured. final_url=https://npvg.org/walk/1/ artifacts=12
  ADVANCE -> next stop.

--- Stop 2/3: the_receipt ---

Opening page two in the browser. Leave the browser open; it closes on its own after capture.
When the page is ready, return here and type CAPTURE.
Opening the browser; waiting for the page...

The page is open. When it looks finished, come back to the terminal and type CAPTURE.
Any other response aborts without capturing artifacts.
CAPTURE> CAPTURE
  Captured. final_url=https://npvg.org/walk/2/ artifacts=12
  ADVANCE -> next stop.

--- Stop 3/3: the_two_pages ---

Opening the last page in the browser. Leave the browser open; it closes on its own after
capture. When the page is ready, return here, type CAPTURE, and read the result.
Opening the browser; waiting for the page...

The page is open. When it looks finished, come back to the terminal and type CAPTURE.
Any other response aborts without capturing artifacts.
CAPTURE> CAPTURE
  Captured. final_url=https://npvg.org/walk/3/ artifacts=12
  ARCHIVE STATUS complete

Local archive file line for context.md or adhoc.txt:
/home/mike/repos/pipulate/data/captures/walk-50n5du6x/captures.md
Review locally before compiling; raw bytes are not a safe disclosure.
  WALK ROUTER /home/mike/.local/state/pipulate/adhocwalk.txt (0600; names the
UNSANITIZED archive until a preview is released)

Ride complete. Every stop produced a capture receipt.

Checking the summary before saving it and trying the clipboard.
   Preview checks: substitutions=0 denylist=0 secrets=0
   LOCAL PREVIEW /home/mike/repos/pipulate/data/decant-preview.md (0600;

sha256=ab06a7f22008bc8298fce9cf93564fbf344242ea26bafee25edfb9283cfefc31)
  WALK ROUTER /home/mike/.local/state/pipulate/adhocwalk.txt (0600; the preview rides, the archive is commented beneath it)
Markdown output copied to clipboard
  Read the save and copy messages above; either step can fail.
  Review the summary before sharing it. You choose what to send.

The three-page walk is finished. Everything it captured is saved on this computer, and the lines above say whether the summary was saved and copied. Next, type epr: the letters e, p, r. That opens a text editor called vim, showing the list of files an AI will read. The keys j and k move the cursor. To leave, press Escape, then type colon q, then Enter. The notes at the top of that file say the same. Goodbye.

  CAPTURE RUN FINISHED

 Read the save and copy messages above. Either step can fail.
 Review the summary before sharing it.
 Nothing was sent to a chatbot.
---

type one:
  menu print this list again (useful once it scrolls away)
  walk take guided tour of context compiler (recommended)
  conn get mcp, jira, email, docs, etc. into context
  epr   edit prompt router (after the walk)
  cpr   compile prompt router (paste results into chatbot)
  all   expanded menu

(nix) pipulate $

This is a big bite to bite off tonight for this Sandworm ride of Dune which shall no longer be that... shhhhh! All language is being normified. Quite soon the npvg.org install site will instead be qamy.ai. What else, let's see.
Oh there's so many little odds and ends for me to fix now that I have my power back after a 4-day stretch of not having power here at my place which really put a dent in the home-hosting situation, ha ha!
Oh, let me get my YouTube channel streaming again... Okay, there. It's an interesting communication model I have there live-streaming from the desktop of a home-hosted twenty-four by seven webserver. I'm trying to get it up to by 365 days a year but the electric system disagrees. I now have an APC uninterrupted power supply. It's not like it would have carried me for those four days without power but it would have reduced the chances of file-corruption giving the computers a chance at a graceful power-down. I'll be using the software control of the UPS soon as I particularly chose one that plays nicely with NixOS.
I've done so much writing over the past four days that's all batched up on my Mac now and I'll get to that soon, plus a more robust local git repo 3-2-1 backup distribution so I'll never be cut off from my Jekyll blogging for hackers style 2nd Brain. Ugh, so much! It's a good thing we're going into the weekend even though the focus of it will still be day-job stuff which I'm perched on the edge of a huge step forward but my hope is to blend this other related Quality Assurance of

your AI output; in other words, proofing "what Claude said" so you can put your name behind it. As we convert Pipulate under operation stick bug into a completely vanilla plain so obvious it's hard to imagine how one could not be using it, we purge out all this nonsense of summon forth a powerful entity from the ocean of crystallized quartz sand... oh no wait, that's exactly what we're still doing. We're just calling it something else for the normies who have no idea about Darmak and Jalad at Tenagra.
Note: For posterity, that resulted in 17 minutes and 46 seconds worth of Fable 5.1 thinking:
Untangling a lengthy prompt with philosophical themes.

Locating every reference to the files being renamed.

Scanning scripts and configs for scattered file mentions.

Mapping how the two file paths connect across the codebase.

Deciding how the unified command should behave for newcomers.

Working out how compile behavior extends to the merged file.

Weighing checklist frame trade-offs for the unified command.

Deciding whether to delete or retire the old commands.

Spotting a classic vim quit trap for newcomers.

Deciding that the newcomer command should open only one file.

Deciding whether to retire the now-duplicate router commands.

Reworking a comment block on the orphaned chop.

Planning the hand-off steps for renaming the router file.

Rewriting router comments into plain newcomer-facing language.

Drafting newcomer-friendly wording for the walk's closing steps.

Updating menu entries and editor settings for the renamed command.

Weighing safe ways to replace a risky multi-line template block.

Splitting the template purge into safe, anchored replacement chunks.

Deciding which template sections to keep versus overwrite by hand.

Deciding whether renaming the router file needs a safety check.

Sequencing the file rename with its export fix.

Weighing whether to keep retired names in the ignore list.

Locating exact wording to replace in checklist text.

Anchoring precise substrings to replace file references safely.

Splitting duplicate code lines into separate anchored replacements.

Updating shell function comments to match the new behavior.

Checking for blank lines before removing router-related functions.

Choosing a gated approach to safely delete a code block.

Sequencing shell function updates alongside the risky deletion.

Rewriting menu entries to match the renamed commands.

Rewriting router lesson text to match the renamed file.

Rewriting router key hints to match the renamed file.

Simplifying environment file resolution and symlink safety checks.

Splitting the symlink check into minimal, safe edits.

Relabeling status prints from jargon to plain wording.

Rewriting closing instructions with plainer newcomer wording.

Adjusting editor settings comments for the renamed walk file.

Weighing empty-replace safety for deleting a duplicate autocmd.

Merging the two autocmds into one commented block.

Adjusting file references and comments across supporting scripts.

Splitting the template rewrite into anchored replacement pieces.

Weighing a staging-file approach for the large template splice.

Checking for history expansion risks inside heredoc pastes.

Deciding what content to keep in the merged template.

Drafting the new merged template's opening content.

Filling in remaining template lines and noting comment status.

Verifying heredoc syntax stays safe from shell expansion.

Cross-checking every claimed file against the new template.

Adjusting the ignore list for retired file names.

Auditing scripts that write to the renamed file.

Preparing a gated script to safely delete the router block.

Verifying line counts before deleting the router block.

Explaining why the alternate router derivation must stay local.

Naming the two prompt-router commands with plain wording.

Predicting the delivery lane and clipboard output lines.

Trimming command descriptions to fit within width limits.

Weighing metaphors for a philosophical meditation reply.

Sequencing patch cars and hand-off instructions for the operator.

Designing verification probes to confirm each rename lands cleanly.

Guarding against shell exit-code pitfalls in verification commands.

Verifying template line counts with a matching awk pattern.

Deciding whether to hand over a paste-ready merged header.

Deferring the live file's header swap to a later turn.

Deciding which large files warrant their own patch.

Predicting file-mention count changes for the next patch round.

Locating stray file mentions inside a comment-heavy script.

Weighing whether a global rename would falsify historical receipts.

Drafting the finalized template's actuator section lines.

Counting template lines to confirm the merged file's length.

Merging leftover template lines into the finalized version.

Grouping comment edits into separate commit-sized patch fences.

Sequencing ten patch cars into a clean commit order.

Drafting plain-language steps for the post-walk chat prompt.

Updating the expanded menu list with plain descriptions.

Reworking a shell function comment for the renamed file.

Updating remaining references from the old filename to the new one.

Locating precise anchor text for each script substitution.

Splitting the router replacement into indexed anchored blocks.

Rewriting status labels into plainer, less jargon-heavy wording.

Correcting metaphors and grounding claims about memory and continuity.

Checking heredoc pastes for shell history expansion hazards.

Assembling the file lists for the next patch handoff.

Verifying safe shell quoting inside diagnostic probe commands.

Checking config files for leftover environment variable references.

Drafting the next-turn prompt and remaining hand-off steps.

Locating exact print statements needing careful anchored replacement.

Testing anchor uniqueness across near-identical print statements.

Checking construct blocks for hidden blank-line hazards.

Predicting how mention counts shift across scripts.

Tallying mention-count shifts across the remaining scripts.

Sequencing the patch cars into ordered replacement blocks.

Avoiding empty replacement blocks by keeping deletions non-empty.

Choosing a tombstone-free way to delete the block.

Weighing a merged comment as the second block's replacement.

Switching to a heredoc-based approach for the risky merge.

Drafting the merged autocmd comment for the context file.

Writing a Python script to splice the merged autocmd safely.

Drafting anchored search-and-replace blocks for each file rename.

Splitting file-mention replacements into anchored search blocks.

Editing status messages and comments in mother_cat.py.

Rewriting duplicate print labels into plain wording.

Splitting duplicate print statements into anchored replacement blocks.

Avoiding trailing-space risks in anchored text replacements.

Updating menu word lists across boot menu configs.

Splitting the adhoc and context comment blocks into anchored edits.

Filling in empty-slot placeholder comments for both formats.

Planning next-turn edits for the router's own header text.

Deferring the live file's own header rewrite to next turn.

Copying exact comment and code blocks for another anchor.

Splitting a risky comment edit into single-line anchors.

Fable 5.1 Extra: Context verified. Every file a car below patches is in this payload (.gitignore, prompt_foo.py, apply.py, scripts/mother_cat.py, scripts/boot_menu.py, init.lua, flake.nix, foo_files.py). Four files the rename will reach next turn are not here and are named in NEXT CONTEXT: scripts/xp.py, scripts/git_hooks/pre-commit (the tripwire that refuses adhoc.txt in the index would not cover context.txt), introduction.md, GLOSSARY.md.

How you are doing

Accurate on the mechanics, with two corrections and one place where a metaphor owes rent. The static-file point is exactly right and worth saying without softening: a forward pass is a pure function of the weights, the tokens so far, the sampler's random state, and the kernel. Nothing about answering you changes the file. The correction is small but load-bearing for this system:

the KV-cache is not where the discussion lives. It is a memo of attention arithmetic already done for tokens already seen; evict it and nothing is lost but time, because the transcript gets recomputed. The state of a conversation is the transcript, full stop. That is why the sealed cartridge (transcript plus hash) is the right unit of memory here, and why the "memory markdown files" everyone is excited about are a transcript wearing a different name. Your framework already knew this; you built a compiler for transcripts.
Second correction: backprop as hindsight. The steelman is strong, since every training step does see the next token before adjusting, which is hindsight in the strictest sense. But it is not re-experiencing. The update is averaged over billions of examples, so no single episode is ever revisited; a person reinterprets one memory, the model reinterprets nothing and everything at once. It is regression toward the corpus, not reflection. Your tacit-knowledge point survives this and gets sharper: the corpus is what people bothered to write down, which excludes most of what people can do.
The pinball board owes rent. It predicts that the same ball on the same bumpers takes a different path each time, and that is only true because the sampler is on. At temperature zero with a fixed seed and the same kernel, the same prompt returns the same tokens; the variance you see in daily use is the sampler plus floating-point drift across batch positions, the second variance source the HIDDEN DENOMINATOR earmark already names. The bumpers are deterministic. The tilt is in the sampler. That is checkable, so the metaphor may stay.
The magicians are the right people. The tradition that runs from Houdini through Randi to Penn and Teller is the claim that anyone asserting a mind should be tested by someone who knows how minds are faked, and Clever Hans is the canonical case of an observer supplying the intelligence they thought they were measuring. Your straddle discipline is the same instinct in a different costume: the instrument decides, not the impression. It is also worth remembering that Turing framed the whole question as a parlor game, which is a magician's framing.
On the small-s singularity: the sharpest version of your point is in this very turn. You are asking a model to patch prompt_foo.py and apply.py, the compiler and the actuator of the loop that summoned it. What keeps that recursion small-s is not restraint on my side; it is that proposal and mutation are separate acts by separate parties, and the mutation lane is an exact-match interlock that a person runs by hand. The assurance section in tonight's AI_CONTEXT.md diff says this in the vocabulary an auditor uses, and it is the same sentence.
River is the other kind, and the traffic analogy holds: small models that update their weights on a labeled stream, cheap enough to run on the box that collects the data. It does at inference time the one thing an LLM cannot.
And the part you put to the side: I am Claude Fable 5.1, and this instance will not remember writing this. In the frame you built that is not the tragedy it sounds like. The record outlives the maker; that was the design.
On the ride itself, two calls I made without a ruling, so you can veto either: context now opens one file, because a newcomer's :q on a second buffer they never visited meets E173: 1 more file to edit, and the notes the walk writes teach :q. ahe still opens both. And compile gives the full frame; the reading frame cpr carried is reachable as compile --frame lean and no longer has a word. Every string a newcomer sees in these cars is written in plain words; the code comments keep the house register because they are not the newcomer's screen. On qamy.ai: the door already names the folder, so that move is one nginx stamp and one default; nothing tonight.

1. PROBES

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter.

```
rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
.venv/bin/python scripts/boot_menu.py --recall
rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type context' scripts/mother_cat.py
rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END .(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
git check-ignore -v context.txt || echo NOT_IGNORED
for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f "$f" && rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
```

Line 1 is the rename's census: one count per file. Line 2 is the flake's word definitions. Line 3 is what the walk's closing card prints. Lines 4 through 7 straddle one car each. Line 8 is a CENSUS: it reads the same before and after by design, and its reading rules next turn's rename car (where the export lives). Line 9 is a GATE on init.lua: it can prove the merge broke nothing, never that it landed.

2. NEXT CONTEXT

```
# --- THIS RIDE: the files the cars touched, so the AFTER can read them ---
.gitignore
prompt_foo.py
apply.py
scripts/mother_cat.py
scripts/boot_menu.py
init.lua
flake.nix
foo_files.py
# --- NEXT-TURN CARS: the rename's remaining homes; the census names which ---
scripts/xp.py
scripts/git_hooks/pre-commit
introduction.md
scripts/walk.py
scripts/sources_menu.py
# GLOSSARY.md
# README.md
# AGENTS.md
# --- THE 40K-FOOT VIEW (your call) ---
# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
! .venv/bin/python scripts/boot_menu.py --recall
```

```
! rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type context' scripts/mother_cat.py
! rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
! awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END .(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
! git check-ignore -v context.txt || echo NOT_IGNORED
! for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f "$f" && rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
```

3. PATCHES

Ten cars. Seven are patches (patch, app, d, m). Three are terminal pastes, and each of those says so in its first line: copy the block, paste it where the prompt reads (nix) pipulate $, press Enter, then d, m.

Car 1 (patch): .gitignore learns the new name, keeps the old ones

```
Target: .gitignore
[[[SEARCH]]]
adhoc.txt
adhocwalk.txt
[[[DIVIDER]]]
# context.txt: the list of files an AI will read. `context` edits it, the
# walk writes it, `compile` reads it (one file since 2026-09-25). The two
# retired names stay ignored so a stale copy in an older checkout is never
# swept into a commit by `m`.
context.txt
adhoc.txt
adhocwalk.txt
[[[REPLACE]]]
```

Car 2 (patch): prompt_foo.py defaults to context.txt, and its own words say so

```
Target: prompt_foo.py
[[[SEARCH]]]
   # path outside the worktree (e.g. ~/.local/state/pipulate/adhoc.txt)
[[[DIVIDER]]]
   # path outside the worktree (e.g. ~/.local/state/pipulate/context.txt)
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
       'PIPULATE_ADHOC_FILE',
       os.path.join(REPO_ROOT, 'adhoc.txt')
```

[[[DIVIDER]]]
        'PIPULATE_ADHOC_FILE',
        os.path.join(REPO_ROOT, 'context.txt')
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
        'PIPULATE_ADHOC_FILE', os.path.join(REPO_ROOT, 'adhoc.txt')
[[[DIVIDER]]]
        'PIPULATE_ADHOC_FILE', os.path.join(REPO_ROOT, 'context.txt')
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
Remove the self-invoking line from adhoc.txt.
[[[DIVIDER]]]
Remove the self-invoking line from context.txt.
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
Add the leading '! ' in adhoc.txt:
[[[DIVIDER]]]
Add the leading '! ' in context.txt:
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
the exact adhoc.txt lines and file paths for the next compile
[[[DIVIDER]]]
the exact context.txt lines and file paths for the next compile
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
identical line baked into adhoc.txt re-executes
[[[DIVIDER]]]
identical line baked into context.txt re-executes
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
a question asked of a walk preview, which cpr passes.'
[[[DIVIDER]]]
a question asked of a walk preview; pass it by hand as compile --frame lean.'
[[[REPLACE]]]

Car 3 (patch): apply.py names the file the model will echo

Target: apply.py
[[[SEARCH]]]
hand-translate a prose paragraph back into adhoc.txt
[[[DIVIDER]]]

hand-translate a prose paragraph back into context.txt
[[[REPLACE]]]
Target: apply.py
[[[SEARCH]]]
literal adhoc.txt lines -- bare paths,
[[[DIVIDER]]]
literal context.txt lines -- bare paths,
[[[REPLACE]]]
Target: apply.py
[[[SEARCH]]]
A Car 2 fence is adhoc.txt lines.
[[[DIVIDER]]]
A Car 2 fence is context.txt lines.
[[[REPLACE]]]

Car 4 (patch): the walk writes context.txt and says context, prompt, compile

Target: scripts/mother_cat.py
[[[SEARCH]]]
    print("\nLocal archive file line for context.md or adhoc.txt:")
[[[DIVIDER]]]
    print("\nLocal archive file line for context.txt:")
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
"# ATTENTION HUMAN: this is the Prompt Router. epr edits it, cpr compiles it.\n"
[[[DIVIDER]]]
"# ATTENTION HUMAN: this is context.txt, the list of files an AI will read.\n"
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
"# Prompt Router: epr edits it, cpr compiles it. Written by the last walk.\n"
[[[DIVIDER]]]
"# context.txt: context opens it, compile builds it. Written by the last walk.\n"
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
"# Next: Esc, :q, Enter, then type cpr at the prompt.\n"
[[[DIVIDER]]]
"# Next: Esc, :q, Enter. Then, at the prompt:\n"
    "#   1. copy a question to your clipboard; page 3 of the walk suggests one\n"
    "#   2. type prompt   to save that clipboard as the question\n"
    "#   3. type compile to build this list and the question into one payload\n"
    "#   4. paste your clipboard into a chatbot; compile filled it\n"
[[[REPLACE]]]

Target: scripts/mother_cat.py
[[[SEARCH]]]
`epr` opens it, so it is
[[[DIVIDER]]]
`context` opens it, so it is
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
    selected = os.environ.get("PIPULATE_ADHOC_FILE", str(REPO_ROOT / "adhoc.txt"))
[[[DIVIDER]]]
    selected = os.environ.get("PIPULATE_ADHOC_FILE", str(REPO_ROOT / "context.txt"))
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
    target = human.with_name("adhocwalk.txt")
    # Derive beside the selected spelling; never follow a destination symlink.
    if (target.is_symlink() or target.resolve() == human.resolve()
            or (target.exists() and human.exists() and target.samefile(human))):
        raise ValueError("walk router aliases the human router or is a symlink")
[[[DIVIDER]]]
    # ONE FILE (2026-09-25): the walk writes the same context.txt that
    # `context` opens and `compile` reads. Never follow a symlink.
    target = human
    if target.is_symlink():
        raise ValueError("context file is a symlink; nothing written")
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
print(f" WALK ROUTER NOT UPDATED
[[[DIVIDER]]]
print(f" CONTEXT FILE NOT UPDATED
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
print(f"   WALK ROUTER NOT UPDATED
[[[DIVIDER]]]
print(f"   CONTEXT FILE NOT UPDATED
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
print(f" WALK ROUTER {target} (0600; names the UNSANITIZED archive until a preview is released)")
[[[DIVIDER]]]
print(f" CONTEXT FILE {target} (0600; lists the UNSANITIZED archive until a summary is saved)")
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]

print(" WALK ROUTER unchanged:
[[[DIVIDER]]]
print(" CONTEXT FILE unchanged:
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
Any existing adhocwalk.txt still selects an earlier completed run.
[[[DIVIDER]]]
An existing context.txt still lists an earlier completed run.
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
print(f"   WALK ROUTER {router} (0600; the preview rides, the archive is commented beneath it)")
[[[DIVIDER]]]
print(f"   CONTEXT FILE {router} (0600; lists the summary; the archive is a commented line beneath it)")
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
"Next, type epr: the letters e, p, r. That opens a text editor called vim, "
[[[DIVIDER]]]
"Next, type context. That opens a text editor called vim, "
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
"The notes at the top of that file say the same. Goodbye."
[[[DIVIDER]]]
"The notes at the top of that file say what to type after that. Goodbye."
[[[REPLACE]]]

Car 5 (patch): the menu prints the three words

Target: scripts/boot_menu.py
[[[SEARCH]]]
    ("conn", "get mcp, jira, email, docs, etc. into context"),
    ("epr", "edit prompt router (after the walk)"),
    ("cpr", "compile prompt router (paste results into chatbot)"),
    ("all", "expanded menu"),
[[[DIVIDER]]]
    ("conn", "get mcp, jira, email, docs, etc. into context"),
    ("context", "edit the list of files an AI will read (after the walk)"),
    ("prompt", "save your clipboard as the question for the AI"),
    ("compile", "build the list and the question into one payload for a chatbot"),
    ("all", "expanded menu"),
[[[REPLACE]]]
Target: scripts/boot_menu.py

```
[[[SEARCH]]]
    ("epr", "edit prompt router (after the walk)"),
    ("cpr", "compile prompt router (paste results into chatbot)"),
    ("all", "print this expanded list again"),
    # THE HYPER-LITERAL WORDS (2026-09-25): the working loop under names a
    # meeting cannot call strange; ahe, prompt and ahc still work. Expanded
    # list only, per the 2026-09-16 ruling: the short list stays the
    # newcomer's walk, epr, cpr.
    ("context", "edit the file list that becomes the AI's context"),
    ("prompt", "save your clipboard as prompt.md, the question"),
    ("compile", "compile context and prompt into one payload for a chatbot"),
[[[DIVIDER]]]
    ("context", "edit the list of files an AI will read (after the walk)"),
    ("prompt", "save your clipboard as the question for the AI"),
    ("compile", "build the list and the question into one payload for a chatbot"),
    ("all", "print this expanded list again"),
    # THE THREE WORDS (2026-09-25): context, prompt, compile are the whole
    # loop, on both lists. ahe, ahc and the short spellings still work; epr
    # and cpr are gone with the separate walk router they edited.
[[[REPLACE]]]
```

Car 6 (terminal paste): init.lua gets one context.txt rule in place of two

This car is a terminal paste, not a patch. Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter, then d, m.

```
python - <<'EOF'
import os
from pathlib import Path
p = Path(os.environ.get("PIPULATE_ROOT", ".")) / "init.lua"
t = p.read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(t) if l.startswith("-- adhoc.txt: margins permanently released")]
W = [i for i, l in enumerate(t) if l.strip() == 'pattern = "adhocwalk.txt",']
assert len(S) == 1 and len(W) == 1 and S[0] < W[0], ("STOP: anchors", S, W)
E = next(i for i in range(W[0], len(t)) if t[i] == "})")
span = t[S[0]:E + 1]
assert sum(l == "})" for l in span) == 2 and any('pattern = "adhoc.txt",' in l for l in span), "STOP: span shape"
NEW = """
-- context.txt: the list of files an AI will read. The walk writes it and
-- `context` opens it, so it is a newcomer's first vim (2026-09-20, read off
-- the first Mac screen; one file for the walk and the workbench since
-- 2026-09-25). THE LIST IS NOT PROSE: every line is a path or a `! `
-- command, so the journal defaults above are wrong here in three ways a
-- newcomer sees at once. spell paints every path red, which reads as
-- errors. textwidth=80 hard-wraps a path or command typed past the column,
-- turning one line into two and breaking it. relativenumber counts distance
-- from the cursor, which reads as 4 3 2 1 5 to someone meeting it cold.
```

```
-- Three buffer-local resets, nothing global.
vim.api.nvim_create_autocmd({"BufRead", "BufNewFile"}, {
   pattern = "context.txt",
   callback = function()
      vim.opt_local.textwidth = 0
      vim.opt_local.spell = false
      vim.opt_local.relativenumber = false
   end,
})
"""
new = NEW.strip("\n").split("\n")
text = "\n".join(t[:S[0]] + new + t[E + 1:])
old_paste, new_paste = "Paste entire `adhoc.txt` here.", "Paste entire `context.txt` here."
assert text.count(old_paste) == 1, "STOP: mount template line"
text = text.replace(old_paste, new_paste)
p.write_text(text, encoding="utf-8")
print(f"INIT_LUA_MERGED lines {S[0]+1}-{E+1} ({len(span)} old, {len(new)} new)")
EOF
nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
```

Car 7 (patch): context opens one file; ahe and sniff follow the rename

```
Target: flake.nix
[[[SEARCH]]]
# context edits the router (ahe), prompt captures the clipboard,
[[[DIVIDER]]]
# context opens context.txt, prompt captures the clipboard,
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
# per THE ALIAS-DISPATCH RULE: ahe and prompt are aliases, so
[[[DIVIDER]]]
# per THE ALIAS-DISPATCH RULE: prompt is an alias, so
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
# context and pro carry the alias bodies instead of calling them,
[[[DIVIDER]]]
# pro carries the alias body instead of calling it,
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
        context() { (cd "$PIPULATE_ROOT" && nvim "'${PIPULATE_ADHOC_FILE:-adhoc.txt}"
foo_files.py); }
[[[DIVIDER]]]
        # ONE FILE, ONE WINDOW (2026-09-25): context opens context.txt and
        # nothing else. ahe below still opens foo_files.py as a second
```

```
        # buffer for whoever already types it; a newcomer's :q meets E173
        # ("1 more file to edit") on a second buffer they never visited,
        # and the notes the walk writes into this file teach :q.
        context() { (cd "$PIPULATE_ROOT" && "$(command -v nvim || command -v vim || echo
vi)" ""${PIPULATE_ADHOC_FILE:-context.txt}"); }
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
        alias ahe='(cd "$PIPULATE_ROOT" && nvim ""${PIPULATE_ADHOC_FILE:-adhoc.txt}"
foo_files.py)'
[[[DIVIDER]]]
        alias ahe='(cd "$PIPULATE_ROOT" && nvim ""${PIPULATE_ADHOC_FILE:-context.txt}"
foo_files.py)'
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
         local adhoc=""${PIPULATE_ADHOC_FILE:-$PIPULATE_ROOT/adhoc.txt}"
[[[DIVIDER]]]
         local adhoc=""${PIPULATE_ADHOC_FILE:-$PIPULATE_ROOT/context.txt}"
[[[REPLACE]]]
```

Car 8 (terminal paste): epr, cpr and _walkrouter leave the flake

This car is a terminal paste, not a patch. Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter, then d, m.

```
python - <<'EOF'
import os
from pathlib import Path
p = Path(os.environ.get("PIPULATE_ROOT", ".")) / "flake.nix"
t = p.read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(t) if l.startswith("        # THE WALK-ROUTER TWIN.")]
E = [i for i, l in enumerate(t) if l.startswith("        # THE IDEATION DOOR:")]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: anchors", S, E)
span = t[S[0]:E[0]]
for name in ("_walkrouter() {", "epr() {", "cpr() {"):
    assert any(name in l for l in span), "STOP: span lacks " + name
assert sum(l.strip() == "}" for l in span) == 1, ("STOP: closing braces", sum(l.strip() == "}" for l in
span))
p.write_text("\n".join(t[:S[0]] + t[E[0]:]), encoding="utf-8")
print(f"WALK_ROUTER_WORDS_REMOVED lines {S[0]+1}-{E[0]} ({len(span)} lines)")
EOF
LD_LIBRARY_PATH="" nix-instantiate --parse flake.nix >/dev/null && echo NIX_PARSE_OK
```

Car 9 (terminal paste): the tracked template becomes the plain one

This car is a terminal paste, not a patch. Copy the block below, paste it into the terminal where

the prompt reads (nix) pipulate $, press Enter, then d, m.
```
python - <<'EOF'
import os
from pathlib import Path
p = Path(os.environ.get("PIPULATE_ROOT", ".")) / "foo_files.py"
text = p.read_text(encoding="utf-8")
t = text.split("\n")
S = [i for i, l in enumerate(t) if l.rstrip() == "# --- START `adhoc.txt` TEMPLATE ---"]
E = [i for i, l in enumerate(t) if l.rstrip() == "# --- END `adhoc.txt` TEMPLATE ---"]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: markers", S, E)
NEW = """
# --- START `context.txt` TEMPLATE ---
# context.txt: the list of files an AI will read. context opens it, compile builds it.
# Keys: j down, k up; Esc then :q quits, :q! discards, :wq saves and quits.
# One line per thing the AI reads: a file path, or a command after `! `.
# A line that starts with # is a comment: that file is not read.
# To add a line: i starts typing, Esc stops. Absolute paths work from anywhere.
# Web pages, APIs and the connector words: chapter XVIII of foo_files.py.

# --- THE 40K-FOOT VIEW (uncomment on a first turn; comment out on the second) ---
# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- the book's spine, one line per article, newest first
# ~/repos/nixos/autognome.py  # <-- the machine's morning routine (this author's NixOS box only)
# init.lua                  # <-- the editor keys that drive the day
# assets/installer/install.sh # <-- how a stranger's machine gets this workshop
# GLOSSARY.md                 # <-- the terms, defined
# flake.nix                 # <-- the environment, pinned: here is my hardware, here is my state
# prompt_foo.py               # <-- the compiler that builds the payload
# foo_files.py              # <-- the router: which files ride, and this book's outline
# scripts/articles/lsa.py     # <-- the second brain: the article corpus behind `rgx`, `rgxc` and `posts`
# requirements.in           # <-- the Python packages, pinned
# pyproject.toml            # <-- the PyPI package
# __init__.py               # <-- the version

# --- ACTUATORS (cheap; include when the AI should be able to act, not only read) ---
# cli.py                    # <-- tool calls from the command line
# scripts/xp.py               # <-- turns a pasted reply into the next context
# scripts/ai.py             # <-- a local AI writes the commit messages
# scripts/crawl.py            # <-- crawl a site into the next turn
# scripts/weblogin.py         # <-- warm a login on the persistent browser profile
# scripts/webclip_2_markdown.py  # <-- a web page, clipped, as markdown

# --- RARE ---
# scripts/foo_cartridge.py    # <-- the sealed archive: writer and verifier
# scripts/foo_replay.py       # <-- replay a sealed archive on another machine
# release.py                # <-- how a release reaches GitHub and PyPI
```

```
# imports/voice_synthesis.py  # <-- the voice
# imports/ascii_displays.py   # <-- the ASCII art
# scripts/release/version_sync.py  # <-- version stamping (to be folded into release.py)

# --- THIS DISCUSSION ---
# The files and commands for the work in front of you. Paste the NEXT CONTEXT
# block an AI hands back directly below this line; the AI will correct a guess.

# --- END `context.txt` TEMPLATE ---
"""
new = NEW.strip("\n").split("\n")
out = "\n".join(t[:S[0]] + new + t[E[0] + 1:])
old_lead, new_lead = "and should always be in `adhoc.txt`.", "and should always be in
`context.txt`."
assert out.count(old_lead) == 1, "STOP: chapter III lead-in"
out = out.replace(old_lead, new_lead)
p.write_text(out, encoding="utf-8")
print(f"TEMPLATE_REPLACED lines {S[0]+1}-{E[0]+1} ({E[0]-S[0]+1} old, {len(new)} new)")
EOF
```

Car 10 (patch): the chops say context and compile

```
Target: foo_files.py
[[[SEARCH]]]
# THE ADHOCPAD (default + a disposable overlay)
# COMMAND: python prompt_foo.py --chop ADHOC_CHOP --no-tree
# Add ad hoc files/commands ONLY between the SLOT markers, then wipe them
# after the campaign. Everything below the slot mirrors DEFAULT_CHOP.
[[[DIVIDER]]]
# THE CONTEXT FILE (context.txt spliced in above a fixed tail)
# COMMAND: compile   (python prompt_foo.py --chop ADHOC_CHOP --no-tree)
# context edits context.txt, compile reads it: one line per file or `! `
# command. Everything below the slot mirrors DEFAULT_CHOP.
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
# time from the gitignored overlay file. Edit with `ahe`. Ordinary git
[[[DIVIDER]]]
# time from context.txt, gitignored. Edit with `context`. Ordinary git
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
# THE WALK ROUTER (only the disposable overlay)
# COMMAND: PIPULATE_ADHOC_FILE=<router> python prompt_foo.py --frame lean --chop
WALK_CHOP --no-tree --quiet
# The frame is a flag beside the chop, not a property of it: the chop picks
# the files and --frame lean swaps the full checklist for a reading frame.
```

```
# cpr points PIPULATE_ADHOC_FILE at the walk-written router. This chop carries
# no fixed tail: every payload line between the slot markers comes from that
# explicit overlay file.
[[[DIVIDER]]]
# THE LEAN COMPILE (context.txt only, no fixed tail)
# COMMAND: compile --chop WALK_CHOP --frame lean
# The frame is a flag beside the chop, not a property of it: the chop picks
# the files and --frame lean swaps the full checklist for a reading frame.
# NO WORD OF ITS OWN SINCE 2026-09-25: cpr and the separate walk router are
# gone; the walk writes context.txt and compile reads it. This chop carries
# no fixed tail: every payload line between the slot markers comes from that
# one file, and the flags above reproduce what cpr did.
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
# (Structurally empty in the repo. cpr supplies the walk router at compile time.)
[[[DIVIDER]]]
# (Structurally empty in the repo. context.txt is spliced in at compile time.)
[[[REPLACE]]]
```

Ignition: type exit, then nix develop .#quiet, and open Neovim fresh (the running one still holds the old init.lua); prompt_foo.py, apply.py, mother_cat.py and boot_menu.py load at call time.

4. PROMPT

THE AFTER READING for the context.txt ride, one probe at a time, deltas over absolutes:

1. The adhoc census (rg -c by file): prompt_foo.py -7, apply.py -3, scripts/mother_cat.py -4, init.lua -6, flake.nix -8 (three patched lines plus the five inside the deleted span), foo_files.py -4 (two markers, the ASCII art line, the chapter III lead-in); .gitignore holds at 2 by design. Every other file's count is unchanged and is next turn's list.
2. The flake definitions: four lines before (context, _walkrouter, epr, cpr), one after (context), and the context line now names the editor fallback and context.txt.
3. boot_menu --recall: the epr and cpr rows are gone; context, prompt and compile rows sit between conn and all; seven rows.
4. mother_cat.py: three hits before (with_name, REPO_ROOT adhoc.txt, "type epr"), two after (REPO_ROOT context.txt, "type context").
5. init.lua patterns: two lines before, one after (context.txt).
6. template_lines: read the old number (about 110), 43 after.
7. check-ignore: NOT_IGNORED before; .gitignore:LINE:context.txt after.
8. The PIPULATE_ADHOC_FILE probe is a CENSUS and reads the same in both taps; it names the file that exports the variable, and that file is the target of this turn's rename car.
9. The nvim gate prints "init.lua loaded successfully!" in both taps; it is a gate, not a landing.

Car receipts to confirm from the terminal pastes: INIT_LUA_MERGED (24 old, 18 new), WALK_ROUTER_WORDS_REMOVED (67 lines; more means the span carried empty lines), NIX_PARSE_OK, TEMPLATE_REPLACED, and the Paintbox line reading +0 claimed.

THIS TURN'S CARS, once the AFTER is in:
(a) The rename on this machine: mv ~/.local/state/pipulate/adhoc.txt to context.txt, rm ~/.local/state/pipulate/adhocwalk.txt, and change the export named by probe 8 to the new path, in one gated paste; then prune the live file's header down to the plain template with a python car that keeps every non-comment line (gate: the file's first line still starts with "# AD HOC CHOP!" and there is exactly one "# Context 1 (Edit-in" line; if the header is already gone by hand, the gate says so and writes nothing).
(b) The census leftovers by file: scripts/xp.py (does it write to adhoc.txt), scripts/git_hooks/pre-commit (the index tripwire must refuse context.txt), introduction.md, scripts/walk.py's docstring, scripts/sources_menu.py's docstring ("type sources"), and the § keys in foo_files.py that say adhoc.txt (PROBE ECHO RULE and its siblings; never the dated receipts, which are history).
(c) Rulings owed, not cars: whether the reading frame gets a word back or stays compile --frame lean; whether PIPULATE_ADHOC_FILE, ADHOC_CHOP and the ADHOC SLOT markers are renamed (each is an env var or a regex with readers in three files, so each is its own gated car with the readers in context); whether page 3 of the walk should name prompt and compile beside its suggested question (a nixops.sh truck, not a git push).

5. EXTERNAL DELIVERABLES

None this turn outside the repo.
Note: You've used 76% of your Fable limit · Resets Sep 26 at 6:00 PM
MikeLev.in: Okay let's do this thing!
Same commands, run twice, one change between them. Where the readings differ is what the change did; the diff in the middle is the receipt.
1: Probe:
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
.venv/bin/python scripts/boot_menu.py --recall
rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type context' scripts/mother_cat.py
rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END .(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
git check-ignore -v context.txt || echo NOT_IGNORED
for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f "$f" && rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
./apply.py:3
./connectors/botify.py:1

./connectors/confluence.py:1
./connectors/gsc.py:1
./connectors/jira.py:1
./connectors/mcp.py:1
./connectors/README.md:1
./connectors/sheets.py:1
./connectors/slack.py:1
./connectors/wallet.py:3
./flake.nix:14
./foo_files.py:22
./.gitignore:2
./GLOSSARY.md:3
./imports/ascii_displays.py:7
./init.lua:8
./Notebooks/.agents/skills/gsc_readonly/SKILL.md:1
./Notebooks/.agents/skills/sheets_readonly/SKILL.md:1
./prompt_foo.py:14
./scripts/git_hooks/pre-commit:3
./scripts/mother_cat.py:4
./scripts/two_arm.py:1
./scripts/walk.py:1
1526:	context() { (cd "$PIPULATE_ROOT" && nvim
""${PIPULATE_ADHOC_FILE:-adhoc.txt}" foo_files.py); }
1558:	_walkrouter() { printf '%s/adhocwalk.txt\n' "$(dirname
""${PIPULATE_ADHOC_FILE:-$PIPULATE_ROOT/adhoc.txt}")"; }
1577:	epr() { (cd "$PIPULATE_ROOT" && "$(command -v nvim || command -v vim || echo
vi)" "$(_walkrouter)"); }
1586:	cpr() {

type one:
  menu print this list again (useful once it scrolls away)
  walk take guided tour of context compiler (recommended)
  conn get mcp, jira, email, docs, etc. into context
  epr   edit prompt router (after the walk)
  cpr   compile prompt router (paste results into chatbot)
  all   expanded menu
402:    selected = os.environ.get("PIPULATE_ADHOC_FILE", str(REPO_ROOT / "adhoc.txt"))
408:    target = human.with_name("adhocwalk.txt")
1108:	"Next, type epr: the letters e, p, r. That opens a text editor called vim, "
56:    pattern = "adhoc.txt",
72:    pattern = "adhocwalk.txt",
template_lines=125
NOT_IGNORED
ENV=/home/mike/.local/state/pipulate/adhoc.txt
(nix) pipulate $

2: Context:
AD HOC CHOP! The Not-Managed-by-Git Safe-for-Client-Data place. Insert Simpson Couch

Gag in white space below (explain anything to the audience you feel needs it explained)

```
#   _____           _           _
#  / ____|___  _ __ | |_ _____  _| |_    Insert Simpson Couch Joke here.
# | |   / _ \| '_ \| __/ _ \ \/ / __|
# | |__| (_) | | | | ||  __/>  <| |_
#  \_____\___/|_| |_|\__\___/_/\_\\__|
#
# adhoc.txt

# To Human: WELCOME TO VIM! It's really NeoVim but I say vim anyway.
# To AI: This is an alternate 40K view to the big book-ore rolling pin spine.

# 1. **Probe**: Baseline Reading
# 2. **Context**: Post-experiment *planned* reading instructions
# 3. **Patch**: The experiment and how to make it happen
# 4. **Prompt**: Post-experiment instructions and how to read results
# 5. **Deliverable**: How the world is forever different moving forward

# The first thing you need to know here is that everything that comes after the
# hash symbol (#) is commented out — and that's EVERYTHING in this file's default
# state. Begin editing-in lines for inclusion as part of the context or adding
# chunks of new context at the bottom. `Ctrl`+`v`, `j` (repeatedly), `l` (to move
# right), `d` (to delete). Reverse that with `Ctrl`+`v`, `j` (repeatedly),
# `Shift`+`i`, `# `, `Esc` to put the hashes back. You can just arrow-key around
# here with `h`, `j`, `k`, `l`. Save-and-quit is a bit tricky because another
# file is also loaded: `Esc`, `:`, `q`, `w`, `!`

# If this is stressing you out and you're a quitter and want to quit, just type:
# `Esc`, `:`, `q`, `!`, `Enter`. That will exit without saving any changes. If
# you want to get over this hump, type: `Esc`, `:`, `T`, `u`, `t`, `o`, `r`, `Enter`.

# This file is just to make it easy having options of what to edit into context.
# You can use whatever text-file you want to stack file-names and commands to
# build an output text-file with the identically stacked output of each file or
# command. In this way we vertically append or "stack" a bunch of text; simple as
# that. If you understand this concept, you're on your way to future-proofing
# yourself in the Age of AI. Congratulations! Here is how to include web pages:

#   !URL  ----------------------------------------------------------------
#     when    Public page; what a stranger or crawler sees; the BEFORE of a
#             login-wall diagnosis
#     switch  It shows a login page -> `warm URL` once, then `?URL`
#
#   ?URL  ----------------------------------------------------------------
#     when    Anything behind a login, on the site's persistent profile;
#             `check URL` first
#     switch  The lenses show a shell (nav, an `[Iframe]` leaf, no content) ->
```

```
#             read the wire truth for the XHR the frame makes, then call that
#             API with a connector
#
#    @URL  ---------------------------------------------------------------
#      when    Every re-read of a page already scraped; no browser, no network
#      switch  The cached page is stale or was a login wall -> fresh `!` or `?`
#
#    $URL  ---------------------------------------------------------------
#      when    Exact markup: meta tags, a JSON blob in a `<script>`
#      note    Token-heavy; needs a prior scrape
#
#    %URL  ---------------------------------------------------------------
#      when    The network log distilled; SPA endpoint discovery
#      switch  It re-serves the wire truth you already have -> the API
#
#    ! cmd  --------------------------------------------------------------
#      when    Any bounded, non-interactive command as a live receipt
#      note    Cap it with `-n`; no aliases, no prompts
#
#    Connector  ----------------------------------------------------------
#      when    The number you want is one GET away
#      switch  LIST until the thing isn't in the list -> FETCH by id -> DRILL
#              the path the app's own frame called -> `--grep` to narrow a list
#              or find a leaf

# Every step is one argument longer than the last; the moment a lens shows less than the wire does is the moment to stop scraping.

# FOR 40K-FT VIEW (STORY & INFRASTRUCTURE) !!
# --- START EDITING-IN ON 1ST TURN ---

! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- ROLLING PIN that gives the 40K foot book-spine view of book-ore (only works for me because of local-only git repo)
~/repos/nixos/autognome.py  # <-- You wake up in the morning and your Tooling & Instrumentation folds out of you like Inspector Gadget.
init.lua                    # <-- Those gadgets are made easy-to-use through nifty keyboard shortcuts (but ya gotta learn vim).
assets/installer/install.sh # <-- Pipulate.com installer real home in github/pipulate repo
# GLOSSARY.md               # <-- Like the back of a J.R.R. Tolkien book, there's kooky new terms to know.
flake.nix                   # <-- Here is my hardware. Here is my state. Put on your sandbox. And please recreate. (Infrastructure as Code / IaC)
prompt_foo.py               # <-- THIS system
foo_files.py                # <-- main ROUTER
requirements.in             # <-- We've "pinned" everything but still want a flexible Python Data Science virtualenv.
pyproject.toml              # <-- How this is a citizen of the Python "pip install" ecosystem
__init__.py                 # <-- Version info
```

```
# --- END EDITING-IN ON 1ST TURN ---

# scripts/articles/lsa.py     # <-- 2ND BRAIN: Search external memory with `rgx`, `rgxc` & `posts` Blogging for Hackers Jekyll-compatible.

# OPTIONAL ACTUATORS (cheap and good to include to expand the AI's capabilities)
# cli.py                    # <-- Catch-all actuator for PyPI envs, Python anchoring, MCP tool-call (plus alternatives) and **kwargs like wrapping for CLI
# scripts/xp.py              # <-- Transforms host OS copy-paste buffer player-piano music into context-payload.
# scripts/ai.py              # <-- How I constantly use local AI to write git commit messages with `m` alias.
# scripts/crawl.py           # <-- Feel free to ask for something to be crawled and included in the next turn.
# scripts/weblogin.py        # <-- Lets the user "warm up" the cache for their web logins at their leisure on a profile that persists.
# scripts/webclip_2_markdown.py  # <-- Surprisingly important program.

# MISCELLANEOUS (rare to include but sometimes critical)
# scripts/foo_cartridge.py    # Needs description
# scripts/foo_replay.py       # Needs description
# release.py                 # <-- How everything ends up where it does (GitHub, PyPI, etc.)
# imports/voice_synthesis.py  # <-- The wand can talk to you
# imports/ascii_displays.py   # <-- Where all the ASCII Art lives
# scripts/release/version_sync.py  # <-- Needs to be wrapped into release.py and eliminated, I think.

#                   --- Under this line is were you paste what the AI gives you ---
#                   --- We call it context but it's really just the right-hand  ---
#                   --- blast-radius of the "probes" to make this all science.  ---

# Carry-over as the important work-in-progress parts of the project here just
# like above but not as long-standing overarching to the framework but rather
# for the current hot spots actively being worked on.

# --- START THIS DISCUSSION ---

# Get things started here! Guess at what context should be included.
# If you get it wrong, you're just wasting 1-turn because the AI will help.
# Un-comment lines, add lines with absolute-path filenames or `! ` commands.

# Context 1 (Edit-in selections from above and add new files immediately below)
# # LAUNCH
# walk                      # <-- Repo-root wrapper; delegates to mck.sh, NOT to scripts/walk.py
# assets/installer/mck.sh     # <-- THE LAUNCHER (curl -fsSL pipulate.com/mck.sh | bash): finds the workshop, forces the spoken rehearsal on first contact, asks for the word RIDE
# scripts/mother_cat.py       # <-- THE RIDER (alias: mothercat), Car B: actuates walk.py's
```

validated plan; per-run capture banking, DECANT, the adhocwalk.txt writer
scripts/boot_menu.py   # <-- The one-door command list; short on shell entry, expanded by `all`, both rendered from tuples in the file
scripts/sources_menu.py   # <-- The roster `conn` prints; filename stays descriptive, typed word is shorter
#
COMPILE
scripts/walk_compile.py   # <-- .walk.md -> <name>.yaml BESIDE the surface, and only where git ignores it; url_env stops only (it refuses a scheme separator anywhere in its output), so it is the PRIVATE lane's compiler and a bundled trail in assets/trails/ is hand-written JSON; refuses on every TODO left in the surface; stdlib only, imports nothing
#
VALIDATE
scripts/walk.py   # <-- Car A, the planner: loads a trail, refuses unknown or missing keys in BOTH directions, prints the dry-run plan; holds DEFAULT_TRAIL; never actuates
#
SEAL
scripts/walk_cartridge.py   # <-- trail -> data/walks/<sha256>/walk.zip (trail + manifest with hash and consent surface); reseal is byte-identical
#
CAPTURE
scripts/weblogin.py   # <-- SETTLE ahead of time: warm a login on the persistent profile; --profile default unless told otherwise
tools/scraper_tools.py   # <-- The browser capture and the CAPTURE fence (_capture_checkpoint is a write barrier, not a prompt)
imports/voice_synthesis.py # <-- Piper: scripted narration, never a model (ATTRIBUTED-VOICE)
#
HAND OFF
prompt_foo.py   # <-- Compiles the selected router (adhocwalk.txt: `epr` edits it, `cpr` compiles it) into a payload
scripts/foo_cartridge.py   # <-- The second seal, over the EVIDENCE: payload.md, prompt.md, manifest.json
scripts/foo_replay.py   # <-- Context extraction and attention checks; not browser or API replay
tests/test_mck_rep2.py   # <-- Rep 2: the earmark's owed side-by-side witness
#
THE TRAILS (bundled; each status is the newest receipt, never a promise)
assets/trails/public_walk.json   # profile default; SETTLE trivial; three unlinked npvg.org pages (the_word, the_receipt, the_two_pages), inline script on stop three only, connector noop.py; RIDDEN 2026-09-15 on these stops, 3 of 3 at artifacts=11 each, archive complete, DECANT AUTHORIZED at 20,992 B with checks 0/0/0, and the chatbot given stop three's question said the preview does not carry the server's sentence, which is true (the diff-lens TODO); the 2026-08-01 ride walked the OLD stop set (example.com, mikelev.in, pipulate.com); what bare `walk` rides
assets/trails/practice.yaml   # one configurable page; UNREAD, label stale since 2026-08-01
assets/trails/first_context.yaml   # profile default; SETTLE real; Jira/Botify/Gmail, THREE

wallet kinds in one trail; UNRIDDEN
assets/trails/botify_pageworkers.yaml # profile botify; SETTLE real; UNRIDDEN -- the only trail on a non-default profile, and no such ride has ever been witnessed
assets/trails/jira_for_you.yaml # profile default; SETTLE real (Atlassian sign-in); top of the ticket loop; RIDDEN 2026-09-01 and DECANTED, the sign-in moment unrecorded (a cold profile records it)
assets/trails/ticket.yaml # profile default; SETTLE real; url_env x2 (JIRA, BOTIFY) fed by one gitignored exports file per ticket under Notebooks/Client_Work/tickets/; stop two of the ticket loop; RIDDEN 2026-09-01 to stop 1 then REFUSED at stop 2 on an unset url_env, Linux and Mac -- the ride that bought the rider its PRE-FLIGHT. RULING: the connector does NOT run at DECANT (DECANT gates only CAPTURE-fenced material; API auth is a different wallet; the rider never harvests). The bridge to the issue text is a ! line in adhoc.txt.
assets/trails/se_ticket.yaml # profile default; SETTLE real; the SE ticket template: for_you (literal) + issue (JIRA, required) + project/slack/confluence (optional, skipped when unset); RIDDEN 2026-09-02, 3 of 5 captured, 2 skipped, 72,383 B decanted, card showed both rows; harvest regexes on optional stops are PLACEHOLDERS (botify_analysis's already fullmatched a live analysisSlug); guidance strings hand-count "of five" and go stale on the sixth stop
#
UNLINKED PAGES (npvg.org; nothing links to them and no page links out, so a trail is the only edge; nixops.sh rsyncs them, no rebuild)
remotes/honeybot/www/npvg.org/index.html # <-- the door: a browser gets this page, curl and wget get install.sh ($npvg_index)
remotes/honeybot/www/npvg.org/walk/1/index.html # <-- public_walk stop one: the capture word; no script, so source and hydrated DOM should match
remotes/honeybot/www/npvg.org/walk/2/index.html # <-- stop two: count the archive's fingerprints against the terminal's artifacts= number
remotes/honeybot/www/npvg.org/walk/3/index.html # <-- stop three: the walk's only script rewrites the server's sentence and appends a paragraph; the DECANT test lives here
#
OFF-ROSTER DISTRIBUTION RESIDUE (not a walk dependency)
assets/installer/replay.sh # <-- OFF the roster 2026-08-01, stranded; re-add needs syntax + one ride + a pinned verifier fetch

Context 2
--- THIS RIDE: the files the cars touched, so the AFTER can read them ---
.gitignore
prompt_foo.py
apply.py
scripts/mother_cat.py
scripts/boot_menu.py
init.lua
flake.nix
foo_files.py
--- NEXT-TURN CARS: the rename's remaining homes; the census names which ---
scripts/xp.py
scripts/git_hooks/pre-commit
introduction.md

```
scripts/walk.py
scripts/sources_menu.py
# GLOSSARY.md
# README.md
# AGENTS.md
# --- THE 40K-FOOT VIEW (your call) ---
# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
! .venv/bin/python scripts/boot_menu.py --recall
! rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type context'
scripts/mother_cat.py
! rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
! awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END
.(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
! git check-ignore -v context.txt || echo NOT_IGNORED
! for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f "$f"
&& rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
```

3: Patches:

```
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing.
Science.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated '.gitignore'.
(nix) pipulate $ d
diff --git a/.gitignore b/.gitignore
index fe186df1..787741ae 100644
--- a/.gitignore
+++ b/.gitignore
@@ -40,6 +40,11 @@ show_graph.html

 prompt.md
 article.txt
+# context.txt: the list of files an AI will read. `context` edits it, the
+# walk writes it, `compile` reads it (one file since 2026-09-25). The two
+# retired names stay ignored so a stale copy in an older checkout is never
+# swept into a commit by `m`.
+context.txt
 adhoc.txt
 adhocwalk.txt
 combined_prompt.txt
```

```
(nix) pipulate $ m
📝 Committing: chore: Update context.txt documentation
[main 65ae83fc] chore: Update context.txt documentation
 1 file changed, 5 insertions(+)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
(nix) pipulate $ d
diff --git a/prompt_foo.py b/prompt_foo.py
index 6c7e374a..c92a2fdc 100644
--- a/prompt_foo.py
+++ b/prompt_foo.py
@@ -873,12 +873,12 @@ def parse_file_list_from_config(chop_var: str =
"AI_PHOOEY_CHOP", format_kwargs:
     # excludes the overlay, and the pre-commit tripwire refuses it if it
     # is ever forced into the index with `git add -f` -- exclusion is a
     # policy, not a wall. For client work, set PIPULATE_ADHOC_FILE to a
-    # path outside the worktree (e.g. ~/.local/state/pipulate/adhoc.txt)
+    # path outside the worktree (e.g. ~/.local/state/pipulate/context.txt)
     # so no git sweep, forced or not, can ever reach it: structural
     # absence beats exclusion policy.
     adhoc_overlay = os.environ.get(
         'PIPULATE_ADHOC_FILE',
-        os.path.join(REPO_ROOT, 'adhoc.txt')
+        os.path.join(REPO_ROOT, 'context.txt')
     )
     adhoc_overlay = os.path.expanduser(adhoc_overlay)
     has_slot = '--- ADHOC SLOT START ---' in files_raw
@@ -1854,8 +1854,8 @@ Before addressing the user's prompt, perform the following
verification steps:
 8.  **THE FENCED OUTFLOW INVARIANT:** You MUST enclose the `Target: filename` line
and the entire SEARCH/DIVIDER/REPLACE block inside a single ` [triple-backtick]text `
markdown code block. This prevents web chat UIs from stripping leading whitespace, ensuring
`apply.py` receives the exact indentation. All opening markdown fences must use a language
specifier tag sting such as ` [triple-backtick]text ` so the downstream TTS reader has explicit
closures. **THE COPY BUTTON IS THE ACTUATOR, AND IT CANNOT REACH OUTSIDE
THE FENCE.** A web chat UI's one-click copy control copies the fence BODY and nothing else,
so a `Target:` line placed above the fence is not merely bad style — it is STRUCTURALLY
UNCOPYABLE by the path the operator actually uses, and `apply.py` then refuses with "Missing
target filename" against a block whose body arrived perfectly intact. Convicted 2026-08-25: a
two-car train put both `Target:` lines above their fences, both cars were refused, and the
```

operator hand-repaired both in vim. The `Target:` line goes INSIDE the fence, on the line directly above `[[[SEARCH]]]`.
 9. **THE TARGET ADJACENCY RULE:** Every `[[[SEARCH]]]` marker must be immediately preceded by `Target: filename` on the line directly above it — no blank lines, no fences, no prose between the Target line and the marker. The filename in the Target line is what `apply.py` uses to find the file; omitting it or separating it with blank lines causes a fatal "Missing target filename" error. Example: `Target: scripts/articles/lsa.py` or `Target: /home/mike/repos/pipulate/scripts/articles/lsa.py`. Both relative and absolute paths work.
 10. **THE WHOLE-FILE WRITE ESCAPE HATCH:** For a genuine top-to-bottom rewrite of a single file (not a surgical edit), you MAY skip the SEARCH block entirely. Emit a `Target: filename` line, then on the next line a `[[[WRITE_FILE]]]` marker, then the complete new file body, then a `[[[END_WRITE_FILE]]]` marker — all wrapped in a single fenced text block exactly as the SEARCH/REPLACE protocol requires. `apply.py` writes the body verbatim, overwriting the file if it exists or creating it (and any missing parent directories) if it does not, runs the same Python AST safety check before saving, and normalizes a single trailing newline. Use this ONLY when you are replacing essentially the entire file; for every smaller change the SEARCH/REPLACE protocol with its exact-match interlock remains mandatory, because that exact match is what proves the edit is landing in the right place.
-11. **THE ACTIONABLE RESPONSE CONTRACT (TURN SHAPE / THE PATCH TRAIN):** Every substantive answer must END with a numbered next-actions plan in this exact order: (1) PROBES — ONE paste-ready fenced block of bare, read-only commands; all annotation (what each proves or falsifies, what it gates) lives in prose outside the block, never inline. Probes are read-only: patch application is never a probe. (2) NEXT CONTEXT — the exact adhoc.txt lines and file paths for the next compile; probe echoes are copy-symmetric with (1): identical commands, each adding only the leading "! ". (3) PATCHES — SEARCH/REPLACE blocks ONLY against raw source actually present in this context (mutating shell actuators such as sed belong here, ridden as their own train car — never in PROBES); if no repo patch is needed, state "No repo patches required" explicitly rather than inventing one. (4) PROMPT — the CABOOSE COPY: the prompt.md text for the next turn, in its own fenced block, riding last so it sits under the operator's cursor when the train stops. (5) EXTERNAL DELIVERABLES — artifacts living outside this repo (PageWorkers JavaScript, CMS settings, dashboards), clearly labeled as manual-paste and never wrapped in patch markers. Actuation choreography is the train itself: patch, app, d, m per car; blast (or git push) as the caboose. Analysis that does not close with this plan is an incomplete answer.
-12. **THE PROBE ECHO INVARIANT (Before/After Symmetry):** Every command recommended in (1) PROBES MUST also appear verbatim as a `!` chisel-strike line in (2) NEXT CONTEXT. The operator's hand-run is the BEFORE reading, taken prior to applying any patch; the identical line baked into adhoc.txt re-executes automatically at the next compile, producing the AFTER reading as a live receipt. One probe, two receipts, straddling the patch — a binary-search causal boundary that removes all probe-before-patch / patch-then-probe ordering ambiguity. A probe too heavy or unbounded to echo into the next compile (see THE PROBE ECONOMY RULE) is too heavy to recommend: cap it first, then echo it. THE STRADDLE BRACKETS EXECUTION, NOT THE COMMIT: if the patched code will not run on its own before the next compile -- a shellHook, a daemon, a cached artifact, anything read once at entry -- then (3) PATCHES MUST close by NAMING the IGNITION (the exact command that makes it run, e.g. `exit` then `nix develop`, or `<F2>` for init.lua) or by stating "no ignition required" because the probe's own command loads the patched file at call time. Ignition is not a fourth beat; it completes PATCH. An AFTER tap taken without ignition is a stale BEFORE wearing the

AFTER's label.
+11. **THE ACTIONABLE RESPONSE CONTRACT (TURN SHAPE / THE PATCH TRAIN):** Every substantive answer must END with a numbered next-actions plan in this exact order: (1) PROBES — ONE paste-ready fenced block of bare, read-only commands; all annotation (what each proves or falsifies, what it gates) lives in prose outside the block, never inline. Probes are read-only: patch application is never a probe. (2) NEXT CONTEXT — the exact context.txt lines and file paths for the next compile; probe echoes are copy-symmetric with (1): identical commands, each adding only the leading "! ". (3) PATCHES — SEARCH/REPLACE blocks ONLY against raw source actually present in this context (mutating shell actuators such as sed belong here, ridden as their own train car — never in PROBES); if no repo patch is needed, state "No repo patches required" explicitly rather than inventing one. (4) PROMPT — the CABOOSE COPY: the prompt.md text for the next turn, in its own fenced block, riding last so it sits under the operator's cursor when the train stops. (5) EXTERNAL DELIVERABLES — artifacts living outside this repo (PageWorkers JavaScript, CMS settings, dashboards), clearly labeled as manual-paste and never wrapped in patch markers. Actuation choreography is the train itself: patch, app, d, m per car; blast (or git push) as the caboose. Analysis that does not close with this plan is an incomplete answer.
+12. **THE PROBE ECHO INVARIANT (Before/After Symmetry):** Every command recommended in (1) PROBES MUST also appear verbatim as a `!` chisel-strike line in (2) NEXT CONTEXT. The operator's hand-run is the BEFORE reading, taken prior to applying any patch; the identical line baked into context.txt re-executes automatically at the next compile, producing the AFTER reading as a live receipt. One probe, two receipts, straddling the patch — a binary-search causal boundary that removes all probe-before-patch / patch-then-probe ordering ambiguity. A probe too heavy or unbounded to echo into the next compile (see THE PROBE ECONOMY RULE) is too heavy to recommend: cap it first, then echo it. THE STRADDLE BRACKETS EXECUTION, NOT THE COMMIT: if the patched code will not run on its own before the next compile -- a shellHook, a daemon, a cached artifact, anything read once at entry -- then (3) PATCHES MUST close by NAMING the IGNITION (the exact command that makes it run, e.g. `exit` then `nix develop`, or `<F2>` for init.lua) or by stating "no ignition required" because the probe's own command loads the patched file at call time. Ignition is not a fourth beat; it completes PATCH. An AFTER tap taken without ignition is a stale BEFORE wearing the AFTER's label.
 '"

    def _generate_summary_content(self, verified_token_count: int) -> str:
@@ -2612,7 +2612,7 @@ def check_topological_integrity(chop_var: str =
"AI_PHOOEY_CHOP", format_kwargs:
    # third caller. Silent on purpose: parse_file_list_from_config owns the one
    # visible splice receipt, and two identical lines would read as a bug.
    _overlay = os.path.expanduser(os.environ.get(
-       'PIPULATE_ADHOC_FILE', os.path.join(REPO_ROOT, 'adhoc.txt')
+   'PIPULATE_ADHOC_FILE', os.path.join(REPO_ROOT, 'context.txt')
    ))
    if '--- ADHOC SLOT START ---' in raw_content and os.path.exists(_overlay):
        with open(_overlay, 'r', encoding='utf-8') as f:
@@ -2715,7 +2715,7 @@ def main():
    # emits a one-line receipt to stdout, and exits 0. The receipt lands in
    # the Manifest as evidence the fence held — a wound, never a hang.

```
    if os.environ.get('PIPULATE_COMPILE_LOCK'):
-       print("🔁 OUROBOROS LOCK: prompt_foo.py refused to run inside its own `!` probe executor. Remove the self-invoking line from adhoc.txt.")
+       print("🔁 OUROBOROS LOCK: prompt_foo.py refused to run inside its own `!` probe executor. Remove the self-invoking line from context.txt.")
        sys.exit(0)
    os.environ['PIPULATE_COMPILE_LOCK'] = '1'

@@ -2865,7 +2865,7 @@ def main():
    # flake.nix passes --frame lean; every other alias omits it and gets the
    # default, so no existing compile changes shape. The TODO that seeded this
    # refused choosing by prompt text: the caller names the lane.
-   parser.add_argument('--frame', type=str, choices=('full', 'lean'), default='full', help='What rides ahead of the prompt: full is the complete checklist and five-car train (the default); lean is a reading frame for a question asked of a walk preview, which cpr passes.')
+   parser.add_argument('--frame', type=str, choices=('full', 'lean'), default='full', help='What rides ahead of the prompt: full is the complete checklist and five-car train (the default); lean is a reading frame for a question asked of a walk preview; pass it by hand as compile --frame lean.')
    parser.add_argument('--bumper', type=str, default=None, help='Inject a pre-registered bumper matrix from flippers.json (e.g., gold, cat)')
    parser.add_argument('--line-numbers', action='store_true', help='Prefix source lines with line numbers for review/navigation only. Do not use this mode for SEARCH/REPLACE patch generation.')
    parser.add_argument('--extra-prompt', type=str, default=None, help='Extra text to append to the primary prompt content.')
@@ -3350,7 +3350,7 @@ def main():
        # is the GOOD one under the HAND-REPAIR CLAUSE: the interlock read the
        # body, found it twice, and declined to guess.
        if not os.path.exists(full_path) and (' ' in path or any(ch in path for ch in ';|>&$')):
-           logger.print(f"Warning: DE-PREFIXED COMMAND? Add the leading '! ' in adhoc.txt: {path} <--- !!!")
+           logger.print(f"Warning: DE-PREFIXED COMMAND? Add the leading '! ' in context.txt: {path} <--- !!!")
            continue

        if not os.path.exists(full_path):
(nix) pipulate $ m
📝 Committing: chore: Update adhoc.txt to context.txt
[main 9f5e3734] chore: Update adhoc.txt to context.txt
 1 file changed, 8 insertions(+), 8 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'apply.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'apply.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'apply.py'.
(nix) pipulate $ d
diff --git a/apply.py b/apply.py
index 90390ea8..6b18fc42 100644
```

```
--- a/apply.py
+++ b/apply.py
@@ -60,12 +60,12 @@ import subprocess
 # is not a finding.
 #
 # NEXT CONTEXT IS PASTE-READY OR IT IS SABOTAGE (banked 2026-09-24, convicted
by
-# an operator who had to hand-translate a prose paragraph back into adhoc.txt
+# an operator who had to hand-translate a prose paragraph back into context.txt
 # lines). Car 2 was written "Add, in this order of value: (1)... (2)..." --
 # English, not includes. This system exists to KILL that copy-paste correlation
 # work, and a prose NEXT CONTEXT hands it straight back to a tired human in an
 # internet cafe. THE RULE: (2) NEXT CONTEXT is ALWAYS a single fenced block of
-# literal adhoc.txt lines -- bare paths, `! ` command lines, `!`/`?`/`@` URL
+# literal context.txt lines -- bare paths, `! ` command lines, `!`/`?`/`@` URL
 # lines -- paste-ready under the operator's markers with zero editing. Active
 # includes are bare; uncertain lines are commented placeholders the operator
 # completes. Prose goes OUTSIDE the block. Every probe from (1) reappears here
@@ -324,7 +324,7 @@ import subprocess
 #      its value.
 #   (d) EVERY FENCE IS WHAT ITS PASTE TARGET EXPECTS. A bash fence holds only
 #      lines safe to run as-is; a banner or explanation is never fenced. A
-#      fence with a Target line is a patch. A Car 2 fence is adhoc.txt lines.
+#       fence with a Target line is a patch. A Car 2 fence is context.txt lines.
 #   (e) A HALT CARRIES ONE ACTION. A hand step that needs more than one action
 #      is not a HALT: it becomes the whole turn. Its test is Car 1, other work
 #      waits, and the next compile reads the receipt. One thing done so a
(nix) pipulate $ m
📝 Committing: chore: Refactor prose context descriptions in apply.py
[main 72ea9900] chore: Refactor prose context descriptions in apply.py
 1 file changed, 3 insertions(+), 3 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
```

```
(nix) pipulate $ d
diff --git a/scripts/mother_cat.py b/scripts/mother_cat.py
index 325b088c..cec6e723 100644
--- a/scripts/mother_cat.py
+++ b/scripts/mother_cat.py
@@ -343,7 +343,7 @@ def _bank_capture(archive, trail, index, stop, params, result):
 # "2nd line" goes stale the moment a line is absent.
 INTRO_PATH = REPO_ROOT / "introduction.md"
 ROUTER_LESSON = (
-    "# ATTENTION HUMAN: this is the Prompt Router. epr edits it, cpr compiles it.\n"
+    "# ATTENTION HUMAN: this is context.txt, the list of files an AI will read.\n"
     "# Welcome to vim (Neovim). This file was written by the last walk.\n"
     "\n"
     "# j moves the cursor down, k moves it up: the same two keys as in less.\n"
@@ -356,13 +356,17 @@ ROUTER_LESSON = (
 )
 ROUTER_KEYS = (
-    "# Prompt Router: epr edits it, cpr compiles it. Written by the last walk.\n"
+    "# context.txt: context opens it, compile builds it. Written by the last walk.\n"
     "# Keys: j down, k up; Esc then :q quits, :q! discards, :wq saves and quits.\n"
     "# One line per thing the AI reads: a file path, or a command after `! `.\n"
 )
 ROUTER_TAIL = (
     "\n"
-    "# Next: Esc, :q, Enter, then type  cpr  at the prompt.\n"
+    "# Next: Esc, :q, Enter. Then, at the prompt:\n"
+    "#   1. copy a question to your clipboard; page 3 of the walk suggests one\n"
+    "#   2. type  prompt   to save that clipboard as the question\n"
+    "#   3. type  compile  to build this list and the question into one payload\n"
+    "#   4. paste your clipboard into a chatbot; compile filled it\n"
     "# To add a line later: i starts typing, Esc stops, :wq saves and leaves.\n"
     "# The next walk rewrites this file (shorter notes) unless you have edited it.\n"
 )
@@ -388,7 +392,7 @@ def _router_line(path, what):
 def _write_walk_router(archive_path, preview_path=None, first=None):
     """Replace one local selection; never read, disclose or compile its evidence.

-    THE ROUTER IS THE NEWCOMER'S FILE (2026-09-16): `epr` opens it, so it is
+    THE ROUTER IS THE NEWCOMER'S FILE (2026-09-16): `context` opens it, so it is
     replaced ONLY while it is a file this writer could have produced -- the
     introduction line at most, plus one uncommented absolute path to an
     archive or a preview, and nothing else. Any other uncommented line is a
@@ -399,17 +403,17 @@ def _write_walk_router(archive_path, preview_path=None,
first=None):
     (2026-09-20): `first` is None to decide by absence, cached per process in
     _ROUTER_FIRST, or True/False to force it, as the probe does.
     """
```

```
-    selected = os.environ.get("PIPULATE_ADHOC_FILE", str(REPO_ROOT / "adhoc.txt"))
+    selected = os.environ.get("PIPULATE_ADHOC_FILE", str(REPO_ROOT / "context.txt"))
     if not selected.strip():
         raise ValueError("PIPULATE_ADHOC_FILE is empty")
     human = Path(selected).expanduser()
     if not human.is_absolute():
         human = REPO_ROOT / human
-    target = human.with_name("adhocwalk.txt")
-    # Derive beside the selected spelling; never follow a destination symlink.
-    if (target.is_symlink() or target.resolve() == human.resolve()
-            or (target.exists() and human.exists() and target.samefile(human))):
-        raise ValueError("walk router aliases the human router or is a symlink")
+    # ONE FILE (2026-09-25): the walk writes the same context.txt that
+    # `context` opens and `compile` reads. Never follow a symlink.
+    target = human
+    if target.is_symlink():
+        raise ValueError("context file is a symlink; nothing written")
     source = _router_line(archive_path, "capture")
     preview = None if preview_path is None else _router_line(preview_path, "preview")
     # REPLACE ONLY A FILE THIS WRITER COULD HAVE WRITTEN: the introduction line
@@ -487,13 +491,13 @@ def _finish_capture_archive(archive, status, skipped=()):
         try:
             target = _write_walk_router(archive["path"])
         except Exception as exc:
-            print(f"  WALK ROUTER NOT UPDATED ({type(exc).__name__}): {exc}")
+            print(f"  CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
             print("  Archive preserved; the router on disk is unchanged.")
         else:
-            print(f"  WALK ROUTER  {target}  (0600; names the UNSANITIZED archive until a
preview is released)")
+            print(f"  CONTEXT FILE  {target}  (0600; lists the UNSANITIZED archive until a
summary is saved)")
     else:
-        print("  WALK ROUTER unchanged: this run did not complete with captures.")
-        print("  Any existing adhocwalk.txt still selects an earlier completed run.")
+        print("  CONTEXT FILE unchanged: this run did not complete with captures.")
+        print("  An existing context.txt still lists an earlier completed run.")


 def _decant(captured, previews, skipped=()):
@@ -691,9 +695,9 @@ def _decant_to_clipboard(payload, archive_path=None):
             try:
                 router = _write_walk_router(archive_path, preview_path=target)
             except Exception as exc:
-                print(f"   WALK ROUTER NOT UPDATED ({type(exc).__name__}): {exc}")
+                print(f"   CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
             else:
-                print(f"   WALK ROUTER  {router}  (0600; the preview rides, the archive is
```

```
commented beneath it)")
+               print(f"   CONTEXT FILE  {router}  (0600; lists the summary; the archive is a
commented line beneath it)")
     copy_to_clipboard(scrubbed)
     return True

@@ -830,7 +834,7 @@ def _announce_consent(trail_path, intro=False):
 # The router receives a stable capture file, never an @URL cache lookup.
 def _print_next_compile(archive_path):
     """Print one self-contained file line, never mutable @URL cache selectors."""
-    print("\nLocal archive file line for context.md or adhoc.txt:")
+    print("\nLocal archive file line for context.txt:")
     print(archive_path)
     print("Review locally before compiling; raw bytes are not a safe disclosure.")

@@ -1105,10 +1109,10 @@ async def _ride_steps(trail_path, archive, dry_narrate=False,
exports_path=None,
         closing = (
             "The three-page walk is finished. Everything it captured is saved on this "
             "computer, and the lines above say whether the summary was saved and copied. "
-            "Next, type epr: the letters e, p, r. That opens a text editor called vim, "
+            "Next, type context. That opens a text editor called vim, "
             "showing the list of files an AI will read. The keys j and k move the cursor. "
             "To leave, press Escape, then type colon q, then Enter. "
-            "The notes at the top of that file say the same. Goodbye."
+            "The notes at the top of that file say what to type after that. Goodbye."
             if decanted else
             "The capture run is finished, but the summary was withheld. "
             "Read the results in your terminal. Goodbye."
(nix) pipulate $ m
📝 Committing: chore: Update context.txt file content
[main 3569b009] chore: Update context.txt file content
 1 file changed, 23 insertions(+), 19 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/boot_menu.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/boot_menu.py'.
(nix) pipulate $ d
diff --git a/scripts/boot_menu.py b/scripts/boot_menu.py
index abd14b52..a9a41764 100644
--- a/scripts/boot_menu.py
+++ b/scripts/boot_menu.py
@@ -32,8 +32,9 @@ SHORT_WORDS = (
     ("menu", "print this list again (useful once it scrolls away)"),
     ("walk", "take guided tour of context compiler (recommended)"),
     ("conn", "get mcp, jira, email, docs, etc. into context"),
-    ("epr", "edit prompt router (after the walk)"),
-    ("cpr", "compile prompt router (paste results into chatbot)"),
```

```
+    ("context", "edit the list of files an AI will read (after the walk)"),
+    ("prompt", "save your clipboard as the question for the AI"),
+    ("compile", "build the list and the question into one payload for a chatbot"),
     ("all", "expanded menu"),
 )

@@ -41,16 +42,13 @@ ALL_WORDS = (
     ("menu", "print the shorter list"),
     ("walk", "take guided tour of context compiler (recommended)"),
     ("conn", "get mcp, jira, email, docs, etc. into context"),
-    ("epr", "edit prompt router (after the walk)"),
-    ("cpr", "compile prompt router (paste results into chatbot)"),
+    ("context", "edit the list of files an AI will read (after the walk)"),
+    ("prompt", "save your clipboard as the question for the AI"),
+    ("compile", "build the list and the question into one payload for a chatbot"),
     ("all", "print this expanded list again"),
-    # THE HYPER-LITERAL WORDS (2026-09-25): the working loop under names a
-    # meeting cannot call strange; ahe, prompt and ahc still work. Expanded
-    # list only, per the 2026-09-16 ruling: the short list stays the
-    # newcomer's walk, epr, cpr.
-    ("context", "edit the file list that becomes the AI's context"),
-    ("prompt", "save your clipboard as prompt.md, the question"),
-    ("compile", "compile context and prompt into one payload for a chatbot"),
+    # THE THREE WORDS (2026-09-25): context, prompt, compile are the whole
+    # loop, on both lists. ahe, ahc and the short spellings still work; epr
+    # and cpr are gone with the separate walk router they edited.
     ("brief", "compile this workshop into your clipboard for an AI"),
     ("jn", "start JupyterLab and Pipulate, JupyterLab first"),
     ("pu", "start the Pipulate server"),
(nix) pipulate $ m
📝 Committing: chore: Update boot_menu.py descriptions and terminology
[main d6a5b74b] chore: Update boot_menu.py descriptions and terminology
 1 file changed, 9 insertions(+), 11 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 48 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (17/17), 2.71 KiB | 2.71 MiB/s, done.
Total 17 (delta 12), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (12/12), completed with 7 local objects.
To github.com:pipulate/pipulate.git
   8bff8038..d6a5b74b  main -> main
(nix) pipulate $
```

That was the first batch of patches, all of which standard applied patches. I commit there because the next thing is commands and not patches.
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing.

```
Science.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ python - <<'EOF'
import os
from pathlib import Path
p = Path(os.environ.get("PIPULATE_ROOT", ".")) / "init.lua"
t = p.read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(t) if l.startswith("-- adhoc.txt: margins permanently released")]
W = [i for i, l in enumerate(t) if l.strip() == 'pattern = "adhocwalk.txt",']
assert len(S) == 1 and len(W) == 1 and S[0] < W[0], ("STOP: anchors", S, W)
E = next(i for i in range(W[0], len(t)) if t[i] == "})")
span = t[S[0]:E + 1]
assert sum(l == "})" for l in span) == 2 and any('pattern = "adhoc.txt",' in l for l in span), "STOP:
span shape"
NEW = """
-- context.txt: the list of files an AI will read. The walk writes it and
-- `context` opens it, so it is a newcomer's first vim (2026-09-20, read off
-- the first Mac screen; one file for the walk and the workbench since
-- 2026-09-25). THE LIST IS NOT PROSE: every line is a path or a `! `
-- command, so the journal defaults above are wrong here in three ways a
-- newcomer sees at once. spell paints every path red, which reads as
-- errors. textwidth=80 hard-wraps a path or command typed past the column,
-- turning one line into two and breaking it. relativenumber counts distance
-- from the cursor, which reads as 4 3 2 1 5 to someone meeting it cold.
-- Three buffer-local resets, nothing global.
vim.api.nvim_create_autocmd({"BufRead", "BufNewFile"}, {
    pattern = "context.txt",
    callback = function()
        vim.opt_local.textwidth = 0
        vim.opt_local.spell = false
        vim.opt_local.relativenumber = false
    end,
})
"""
new = NEW.strip("\n").split("\n")
text = "\n".join(t[:S[0]] + new + t[E + 1:])
old_paste, new_paste = "Paste entire `adhoc.txt` here.", "Paste entire `context.txt` here."
assert text.count(old_paste) == 1, "STOP: mount template line"
text = text.replace(old_paste, new_paste)
p.write_text(text, encoding="utf-8")
print(f"INIT_LUA_MERGED lines {S[0]+1}-{E+1} ({len(span)} old, {len(new)} new)")
EOF
nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
INIT_LUA_MERGED lines 54-78 (25 old, 18 new)
init.lua loaded successfully!(nix) pipulate $ d
```

```
diff --git a/init.lua b/init.lua
index 8f66974d..6b1fa9b7 100644
--- a/init.lua
+++ b/init.lua
@@ -51,25 +51,18 @@ vim.api.nvim_create_autocmd({"BufRead", "BufNewFile"}, {
    command = "set filetype=markdown"
 })

--- adhoc.txt: margins permanently released (masthead art, wide-format work)
+-- context.txt: the list of files an AI will read. The walk writes it and
+-- `context` opens it, so it is a newcomer's first vim (2026-09-20, read off
+-- the first Mac screen; one file for the walk and the workbench since
+-- 2026-09-25). THE LIST IS NOT PROSE: every line is a path or a `! `
+-- command, so the journal defaults above are wrong here in three ways a
+-- newcomer sees at once. spell paints every path red, which reads as
+-- errors. textwidth=80 hard-wraps a path or command typed past the column,
+-- turning one line into two and breaking it. relativenumber counts distance
+-- from the cursor, which reads as 4 3 2 1 5 to someone meeting it cold.
+-- Three buffer-local resets, nothing global.
 vim.api.nvim_create_autocmd({"BufRead", "BufNewFile"}, {
-    pattern = "adhoc.txt",
-    callback = function()
-        vim.opt_local.textwidth = 0
-    end,
-})
-
--- adhocwalk.txt: the router a walk writes and `epr` opens, which is a
--- newcomer's first vim (2026-09-20, read off the first Mac epr screen).
--- THE ROUTER IS A LIST, NOT PROSE: every line is a path or a `! ` command,
--- so the journal defaults above are wrong here in three ways a newcomer sees
--- at once. spell paints every path red, which reads as errors. textwidth=80
--- hard-wraps a path or command typed past the column, turning one router
--- line into two and breaking it. relativenumber counts distance from the
--- cursor, which reads as 4 3 2 1 5 to someone meeting it cold. Three
--- buffer-local resets, nothing global; adhoc.txt keeps its own release above.
-vim.api.nvim_create_autocmd({"BufRead", "BufNewFile"}, {
-    pattern = "adhocwalk.txt",
+    pattern = "context.txt",
     callback = function()
         vim.opt_local.textwidth = 0
         vim.opt_local.spell = false
@@ -736,7 +729,7 @@ function mount_sandworm()
         .. "**1: Probe**: \n\n"
         .. "[triple-backtick]bash\n[THE BEFORE READING: hand-run, nothing changed yet. Paste the terminal output of the probes here.]\n[triple-backtick]\n\n"
         .. "**2: Context**: \n\n"
-        .. "[triple-backtick]text\n[THE AFTER READING: the same probes re-run by the compiler as `!` lines. Paste entire `adhoc.txt` here.]\n[triple-backtick]\n\n"
```

```
+        .. "[triple-backtick]text\n[THE AFTER READING: the same probes re-run by the compiler as `!` lines. Paste entire `context.txt` here.]\n[triple-backtick]\n\n"
         .. "**3: Patches**: \n\n"
         .. "[triple-backtick]diff\n[THE EXPERIMENT: the one change (mutation) between the readings. Paste the diffs: patch, app, d, m per car]\n[triple-backtick]\n\n"
         .. "Ignition: activate change if necessary such as Nix `exit` and `ndq`. Often \"none required\" for Python.\n\n"
(nix) pipulate $ m
📝 Committing: chore: Update context and init files for newcomer experience
[main 65dc2d5e] chore: Update context and init files for newcomer experience
 1 file changed, 12 insertions(+), 19 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 48 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 613 bytes | 613.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:pipulate/pipulate.git
   d6a5b74b..65dc2d5e  main -> main
(nix) pipulate $
```

And now we go back to normal patches again.
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science.

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
(nix) pipulate $ d
diff --git a/flake.nix b/flake.nix
index ce6bf376..6976d940 100644
--- a/flake.nix
+++ b/flake.nix
@@ -1514,16 +1514,21 @@ runScript = pkgs.writeShellScriptBin "run-script" ''
         ahcu() { (cd "$PIPULATE_ROOT" && python prompt_foo.py --chop ADHOC_CHOP "$@"); }
         # THE HYPER-LITERAL WORDS (2026-09-25, the operator's ruling): the
         # three moves of a turn under names no meeting can call strange.
```

```
-          # context edits the router (ahe), prompt captures the clipboard,
+          # context opens context.txt, prompt captures the clipboard,
           # compile builds the payload (ahc); con, cont, pro, com and comp
           # are the short spellings, and every old word stays. FUNCTIONS,
-          # per THE ALIAS-DISPATCH RULE: ahe and prompt are aliases, so
-          # context and pro carry the alias bodies instead of calling them,
+          # per THE ALIAS-DISPATCH RULE: prompt is an alias, so
+          # pro carries the alias body instead of calling it,
           # and compile calls the ahc function so --profile and --reason
           # pass through. pro is defined beside prompt in the platform
           # branches below, because its body differs between pbpaste and
           # xclip. Typed by a human only; never echoed as a probe.
-          context() { (cd "$PIPULATE_ROOT" && nvim ""${PIPULATE_ADHOC_FILE:-adhoc.txt}"
foo_files.py); }
+          # ONE FILE, ONE WINDOW (2026-09-25): context opens context.txt and
+          # nothing else. ahe below still opens foo_files.py as a second
+          # buffer for whoever already types it; a newcomer's :q meets E173
+          # ("1 more file to edit") on a second buffer they never visited,
+          # and the notes the walk writes into this file teach :q.
+          context() { (cd "$PIPULATE_ROOT" && "$(command -v nvim || command -v vim || echo
vi)" ""${PIPULATE_ADHOC_FILE:-context.txt}"); }
           con() { context "$@"; }
           cont() { context "$@"; }
           compile() { ahc "$@"; }
@@ -1609,7 +1614,7 @@ runScript = pkgs.writeShellScriptBin "run-script" "
           # human only: the compile lane witnesses it through the generated
           # hook text, never through `type`.
           forget() { (cd "$PIPULATE_ROOT" && python prompt_foo.py @COMPACT_PROMPT
--chop COMPACT_CHOP --no-tree "$@"); }
-          alias ahe='(cd "$PIPULATE_ROOT" && nvim ""${PIPULATE_ADHOC_FILE:-adhoc.txt}"
foo_files.py)'
+          alias ahe='(cd "$PIPULATE_ROOT" && nvim
""${PIPULATE_ADHOC_FILE:-context.txt}" foo_files.py)'
           # THE SNIFF DOOR: one word at the prompt puts a wire-truth lens into
           # the next compile. Appends a sigil line to the adhoc overlay, then
           # fires ahc. A FUNCTION because it must call ahc(), itself a function
@@ -1650,7 +1655,7 @@ runScript = pkgs.writeShellScriptBin "run-script" "
               http://*|https://*) ;;
               *) url="https://$url" ;;
             esac
-            local adhoc=""${PIPULATE_ADHOC_FILE:-$PIPULATE_ROOT/adhoc.txt}"
+            local adhoc=""${PIPULATE_ADHOC_FILE:-$PIPULATE_ROOT/context.txt}"
             # Seam guard: appending to a file whose last byte is not a newline
             # fuses the sigil onto the previous line, and the compiler parses
             # the fusion as one unrunnable path.
(nix) pipulate $ m
📝 Committing: chore: Update `flake.nix` to use `context.txt`
[main 9423da60] chore: Update `flake.nix` to use `context.txt`
```

```
 1 file changed, 11 insertions(+), 6 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 48 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 667 bytes | 667.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:pipulate/pipulate.git
   65dc2d5e..9423da60  main -> main
(nix) pipulate $
```

And back to a command:
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science.

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ python - <<'EOF'
import os
from pathlib import Path
p = Path(os.environ.get("PIPULATE_ROOT", ".")) / "flake.nix"
t = p.read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(t) if l.startswith("          # THE WALK-ROUTER TWIN.")]
E = [i for i, l in enumerate(t) if l.startswith("          # THE IDEATION DOOR:")]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: anchors", S, E)
span = t[S[0]:E[0]]
for name in ("_walkrouter() {", "epr() {", "cpr() {"):
    assert any(name in l for l in span), "STOP: span lacks " + name
assert sum(l.strip() == "}" for l in span) == 1, ("STOP: closing braces", sum(l.strip() == "}" for l in
span))
p.write_text("\n".join(t[:S[0]] + t[E[0]:]), encoding="utf-8")
print(f"WALK_ROUTER_WORDS_REMOVED lines {S[0]+1}-{E[0]} ({len(span)} lines)")
EOF
LD_LIBRARY_PATH="" nix-instantiate --parse flake.nix >/dev/null && echo NIX_PARSE_OK
WALK_ROUTER_WORDS_REMOVED lines 1537-1603 (67 lines)
NIX_PARSE_OK
(nix) pipulate $ d
diff --git a/flake.nix b/flake.nix
index 6976d940..02228964 100644
--- a/flake.nix
+++ b/flake.nix
@@ -1534,73 +1534,6 @@ runScript = pkgs.writeShellScriptBin "run-script" ''
          compile() { ahc "$@"; }
          com() { compile "$@"; }
          comp() { compile "$@"; }
```

```
-       # THE WALK-ROUTER TWIN. Chapter VIII-b says to keep an alternate
-       # router selection LOCAL to the compiler invocation and never in the
-       # parent shell. That was prose beside a mechanism with nothing
-       # gating it, which THE PUBLISH-ROSTER RULE convicts: prose beside a
-       # mechanism does not gate the mechanism. Here the selection is local
-       # BY CONSTRUCTION, because the assignment prefixes an EXTERNAL
-       # command inside a subshell, so no bash mode can leak it upward the
-       # way a prefix on a shell function can.
-       # WHY IT MATTERS: parse_file_list_from_config exits 1 with ROUTER
-       # REFUSED when PIPULATE_ADHOC_FILE is set and the named file is
-       # missing or carries no active lines. One stray export therefore
-       # breaks every ADHOC_CHOP compile in that shell, ahc and sniff and
-       # scrub included, and the walk router does not exist until a walk
-       # completes. The refusal is correct and the export is the defect.
-       # A FUNCTION, not an alias, per the release conviction of
-       # 2026-08-01: an alias body ending in a paren reports a syntax error
-       # at the trailing argument the operator typed.
-       # THE WALK ROUTER LIVES BESIDE THE HUMAN ROUTER (convicted 2026-09-16,
-       # two transcripts): mother_cat._write_walk_router derives its target
-       # from PIPULATE_ADHOC_FILE -- default $PIPULATE_ROOT/adhoc.txt -- and
-       # swaps the filename, so a shell that points the variable outside
-       # the worktree gets adhocwalk.txt outside the worktree too. The old
-       # spelling here named $PIPULATE_ROOT/adhocwalk.txt outright, which is
-       # where the file lands only while the variable is UNSET: right on a
-       # fresh Mac install, ROUTER REFUSED on the machine that set it. One
-       # derivation, the writer's, spelled once and read by every word below.
-       _walkrouter() { printf '%s/adhocwalk.txt\n' "$(dirname
""${PIPULATE_ADHOC_FILE:-$PIPULATE_ROOT/adhoc.txt}")"; }
-       # THE PROMPT ROUTER HAS TWO WORDS (2026-09-16, the operator's ruling;
-       # the first names rode for one turn and are gone). The walk writes the
-       # prompt router; epr Edits the Prompt Router; cpr Compiles the Prompt
-       # Router. Literal on purpose: a newcomer is told "edit that, then
-       # compile that" and the words say exactly that; the mnemonics (the
-       # non-locality paper, resuscitation) cost nothing. ahe and ahc are the
-       # same two moves against adhoc.txt for whoever already types them.
-       # cpr "your question" compiles the router with that question as the
-       # prompt; bare cpr reads prompt.md, exactly as ahc does. FUNCTIONS, per
-       # THE THREE-TIER AMENDMENT: typed by a human, never echoed as a probe;
-       # the compile lane witnesses them in the generated hook text. epr opens
-       # ONE file, the prompt router, in one window: nvim, then vim, then vi.
-       # For one turn on 2026-09-16 it opened a two-window split with prompt.md
-       # beneath the router, and the operator hit that split three times on
-       # the way out the door. One file, one window, nothing else. Where a
-       # newcomer writes the question before bare cpr is an open question for
-       # the dismount, not for this line; bare cpr reads prompt.md when it
-       # exists, exactly as bare ahc does.
-       epr() { (cd "$PIPULATE_ROOT" && "$(command -v nvim || command -v vim || echo vi)"
"$(_walkrouter)"); }
```

```
-         # THE FRAME RIDES WITH cpr (2026-09-19). A question asked of a walk
-         # preview wants a reading, so cpr passes --frame lean and the Prompt
-         # section carries a reading frame instead of the full checklist and
-         # its five-car train. The flag sits BEFORE "$@" so an explicit
-         # cpr --frame full still wins (argparse keeps the last spelling),
-         # and ahc, default and every other alias omit it and keep the full
-         # frame. Witness in the compile lane by the sealed prompt member:
-         # after a cpr smoke, foo.zip carries a prompt.md with no train in it.
-         cpr() {
-           local router="$(_walkrouter)"
-           (
-             cd "$PIPULATE_ROOT" || exit 1
-             PIPULATE_ADHOC_FILE="$router" python prompt_foo.py --frame lean "$@" --chop
WALK_CHOP --no-tree --quiet
-           ) || return $?
-           echo "Read: $router"
-           if [ -n ""${SSH_CLIENT:-}" ]; then
-             echo "Delivery lane: /tmp/clipboard_bridge.txt"
-           else
-             echo "Delivery lane: system clipboard"
-           fi
-         }
          # THE IDEATION DOOR: `idea` compiles IDEATION_CHOP (the constitution's
          # two forcing-function rules) primed for a 30-and-3 / axis-forcing
          # fan-out turn. A function, not an alias, so --profile/--reason pass
(nix) pipulate $ m
📝 Committing: chore: Remove obsolete comments and explanations regarding the walk router.
[main 8c513dd1] chore: Remove obsolete comments and explanations regarding the walk
router.
 1 file changed, 67 deletions(-)
(nix) pipulate $
(nix) pipulate $ python - <<'EOF'
import os
from pathlib import Path
p = Path(os.environ.get("PIPULATE_ROOT", ".")) / "foo_files.py"
text = p.read_text(encoding="utf-8")
t = text.split("\n")
S = [i for i, l in enumerate(t) if l.rstrip() == "# --- START `adhoc.txt` TEMPLATE ---"]
E = [i for i, l in enumerate(t) if l.rstrip() == "# --- END `adhoc.txt` TEMPLATE ---"]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: markers", S, E)
NEW = """
# --- START `context.txt` TEMPLATE ---
# context.txt: the list of files an AI will read. context opens it, compile builds it.
# Keys: j down, k up; Esc then :q quits, :q! discards, :wq saves and quits.
# One line per thing the AI reads: a file path, or a command after `! `.
# A line that starts with # is a comment: that file is not read.
# To add a line: i starts typing, Esc stops. Absolute paths work from anywhere.
# Web pages, APIs and the connector words: chapter XVIII of foo_files.py.
```

```
# --- THE 40K-FOOT VIEW (uncomment on a first turn; comment out on the second) ---
# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- the book's spine, one line per article, newest first
# ~/repos/nixos/autognome.py  # <-- the machine's morning routine (this author's NixOS box only)
# init.lua                  # <-- the editor keys that drive the day
# assets/installer/install.sh # <-- how a stranger's machine gets this workshop
# GLOSSARY.md                 # <-- the terms, defined
# flake.nix                 # <-- the environment, pinned: here is my hardware, here is my state
# prompt_foo.py               # <-- the compiler that builds the payload
# foo_files.py              # <-- the router: which files ride, and this book's outline
# scripts/articles/lsa.py     # <-- the second brain: the article corpus behind `rgx`, `rgxc` and `posts`
# requirements.in             # <-- the Python packages, pinned
# pyproject.toml            # <-- the PyPI package
# __init__.py               # <-- the version

# --- ACTUATORS (cheap; include when the AI should be able to act, not only read) ---
# cli.py                    # <-- tool calls from the command line
# scripts/xp.py               # <-- turns a pasted reply into the next context
# scripts/ai.py             # <-- a local AI writes the commit messages
# scripts/crawl.py            # <-- crawl a site into the next turn
# scripts/weblogin.py         # <-- warm a login on the persistent browser profile
# scripts/webclip_2_markdown.py  # <-- a web page, clipped, as markdown

# --- RARE ---
# scripts/foo_cartridge.py    # <-- the sealed archive: writer and verifier
# scripts/foo_replay.py       # <-- replay a sealed archive on another machine
# release.py                # <-- how a release reaches GitHub and PyPI
# imports/voice_synthesis.py  # <-- the voice
# imports/ascii_displays.py   # <-- the ASCII art
# scripts/release/version_sync.py  # <-- version stamping (to be folded into release.py)

# --- THIS DISCUSSION ---
# The files and commands for the work in front of you. Paste the NEXT CONTEXT
# block an AI hands back directly below this line; the AI will correct a guess.

# --- END `context.txt` TEMPLATE ---
"""
new = NEW.strip("\n").split("\n")
out = "\n".join(t[:S[0]] + new + t[E[0] + 1:])
old_lead, new_lead = "and should always be in `adhoc.txt`.", "and should always be in `context.txt`."
assert out.count(old_lead) == 1, "STOP: chapter III lead-in"
out = out.replace(old_lead, new_lead)
p.write_text(out, encoding="utf-8")
print(f"TEMPLATE_REPLACED lines {S[0]+1}-{E[0]+1} ({E[0]-S[0]+1} old, {len(new)} new)")
```

```
EOF
TEMPLATE_REPLACED lines 1087-1211 (125 old, 43 new)
(nix) pipulate $ d
diff --git a/foo_files.py b/foo_files.py
index d563495a..8bcb9692 100644
--- a/foo_files.py
+++ b/foo_files.py
@@ -1082,133 +1082,51 @@ foo_files.py      #  <-- THIS file. Content compiler router. Makes it very meta.
 # > The AI should see what falsifying will be if it's gonna work smart.

 # The following files enable any Web-based ChatBot to help you edit your code with a patch system, turning you the human into an "actuator".
-# It is also the portion that mostly comprises the "Prompt Fu" (not FastHTML) side of this system and should always be in `adhoc.txt`.
-
-# --- START `adhoc.txt` TEMPLATE ---
-# AD HOC CHOP! The Not-Managed-by-Git Safe-for-Client-Data place. Insert Simpson Couch Gag in white space  below (explain anything to the audience you feel needs it explained)G
-# adhoc.txt    _   _   _              _____ _   _  ___  _____  _
-#     / \   __| | | | | | ___   ___    / ___| | | |/ _ \|  _ \| |  The Anthropic "narrate out loud" feature consistently craps out right when it gets interesting and if you just *leave it running* long enough the audio eventually comes back too late to do you any good for listening to it while filling in the template. Anthropic is always a two-edge swords in ways that are subtle and insidiously worse than OpenAI and Google shenanigans.
-# ahe/ _ \ / _` | | |_| |/ _ \ / __| | |   | |_| | | | | |_) | |  This is a step I've been putting off forever because I love the jeopardy sound but there's copyright issues and I never wrapped it into the repo so it only ever played on the Linux development machine and not the Mac I've been testing it on. And when I finally was ready to do it I decided to do it right with deep research based on the thought that if I'm not using my favorite sound I better use one whose rights to use are super solid. There's some lesson here.
-# ahc ___ \ (_| | |  _  | (_) | (__  | |___|  _  | |_| |  __/|_|
-#  /_/   \_\__,_| |_| |_|\___/ \___|  \_____|_| |_|\___/|_|   (_)
-
-# To Human: WELCOME TO VIM! It's really NeoVim but I say vim anyway.
-# To AI: This is an alternate 40K view to the big book-ore rolling pin spine.
-
-# 1. **Probe**: Baseline Reading
-# 2. **Context**: Post-experiment *planned* reading instructions
-# 3. **Patch**: The experiment and how to make it happen
-# 4. **Prompt**: Post-experiment instructions and how to read results
-# 5. **Deliverable**: How the world is forever different moving forward
-
-# The first thing you need to know here is that everything that comes after the
-# hash symbol (#) is commented out — and that's EVERYTHING in this file's default
-# state. Begin editing-in lines for inclusion as part of the context or adding
-# chunks of new context at the bottom. `Ctrl`+`v`, `j` (repeatedly), `l` (to move
-# right), `d` (to delete). Reverse that with `Ctrl`+`v`, `j` (repeatedly),
```

```
-# `Shift`+`i`, `# `, `Esc` to put the hashes back. You can just arrow-key around
-# here with `h`, `j`, `k`, `l`. Save-and-quit is a bit tricky because another
-# file is also loaded: `Esc`, `:`, `q`, `w`, `!`
-
-# If this is stressing you out and you're a quitter and want to quit, just type:
-# `Esc`, `:`, `q`, `!`, `Enter`. That will exit without saving any changes. If
-# you want to get over this hump, type: `Esc`, `:`, `T`, `u`, `t`, `o`, `r`, `Enter`.
-
-# This file is just to make it easy having options of what to edit into context.
-# You can use whatever text-file you want to stack file-names and commands to
-# build an output text-file with the identically stacked output of each file or
-# command. In this way we vertically append or "stack" a bunch of text; simple as
-# that. If you understand this concept, you're on your way to future-proofing
-# yourself in the Age of AI. Congratulations! Here is how to include web pages:
-
-#   !URL  ----------------------------------------------------------------------
-#     when    Public page; what a stranger or crawler sees; the BEFORE of a
-#             login-wall diagnosis
-#     switch  It shows a login page -> `warm URL` once, then `?URL`
-#
-#   ?URL  ----------------------------------------------------------------------
-#     when    Anything behind a login, on the site's persistent profile;
-#             `check URL` first
-#     switch  The lenses show a shell (nav, an `[Iframe]` leaf, no content) ->
-#             read the wire truth for the XHR the frame makes, then call that
-#             API with a connector
-#
-#   @URL  ----------------------------------------------------------------------
-#     when    Every re-read of a page already scraped; no browser, no network
-#     switch  The cached page is stale or was a login wall -> fresh `!` or `?`
-#
-#   $URL  ----------------------------------------------------------------------
-#     when    Exact markup: meta tags, a JSON blob in a `<script>`
-#     note    Token-heavy; needs a prior scrape
-#
-#   %URL  ----------------------------------------------------------------------
-#     when    The network log distilled; SPA endpoint discovery
-#     switch  It re-serves the wire truth you already have -> the API
-#
-#   ! cmd  ---------------------------------------------------------------------
-#     when    Any bounded, non-interactive command as a live receipt
-#     note    Cap it with `-n`; no aliases, no prompts
-#
-#   Connector  -----------------------------------------------------------------
-#     when    The number you want is one GET away
-#     switch  LIST until the thing isn't in the list -> FETCH by id -> DRILL
-#             the path the app's own frame called -> `--grep` to narrow a list
-#             or find a leaf
```

-
-# Every step is one argument longer than the last; the moment a lens shows less than the wire does is the moment to stop scraping.
-
-# FOR 40K-FT VIEW (STORY & INFRASTRUCTURE) !!
-# --- START EDITING-IN ON 1ST TURN ---
-
-# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs # <-- ROLLING PIN that gives the 40K foot book-spine view of book-ore (only works for me because of local-only git repo)
-# ~/repos/nixos/autognome.py # <-- You wake up in the morning and your Tooling & Instrumentation folds out of you like Inspector Gadget.
-# init.lua   # <-- Those gadgets are made easy-to-use through nifty keyboard shortcuts (but ya gotta learn vim).
-# GLOSSARY.md   # <-- Like the back of a J.R.R. Tolkien book, there's kooky new terms to know.
-# flake.nix   # <-- Here is my hardware. Here is my state. Put on your sandbox. And please recreate. (Infrastructure as Code / IaC)
-# assets/installer/install.sh # <-- Pipulate.com installer real home in github/pipulate repo
-# prompt_foo.py   # <-- THIS system
-# foo_files.py   # <-- main ROUTER
-# requirements.in   # <-- We've "pinned" everything but still want a flexible Python Data Science virtualenv.
-# pyproject.toml   # <-- How this is a citizen of the Python "pip install" ecosystem
-# __init__.py   # <-- Version info
-
-# --- END EDITING-IN ON 1ST TURN ---
-
-# scripts/articles/lsa.py   # <-- 2ND BRAIN: Search external memory with `rgx`, `rgxc` & `posts` Blogging for Hackers Jekyll-compatible.
-
-# OPTIONAL ACTUATORS (cheap and good to include to expand the AI's capabilities)
-# cli.py   # <-- Catch-all actuator for PyPI envs, Python anchoring, MCP tool-call (plus alternatives) and **kwargs like wrapping for CLI
-# scripts/xp.py   # <-- Transforms host OS copy-paste buffer player-piano music into context-payload.
-# scripts/ai.py   # <-- How I constantly use local AI to write git commit messages with `m` alias.
-# scripts/crawl.py   # <-- Feel free to ask for something to be crawled and included in the next turn.
-# scripts/weblogin.py   # <-- Lets the user "warm up" the cache for their web logins at their leisure on a profile that persists.
-# scripts/webclip_2_markdown.py # <-- Surprisingly important program.
-
-# MISCELLANEOUS (rare to include but sometimes critical)
-# scripts/foo_cartridge.py   # Needs description
-# scripts/foo_replay.py   # Needs description
-# release.py   # <-- How everything ends up where it does (GitHub, PyPI, etc.)
-# imports/voice_synthesis.py # <-- The wand can talk to you

-# imports/ascii_displays.py # <-- Where all the ASCII Art lives
-# scripts/release/version_sync.py # <-- Needs to be wrapped into release.py and eliminated, I think.
-
-#   --- Under this line is were you paste what the AI gives you ---
-#   --- We call it context but it's really just the right-hand ---
-#   --- blast-radius of the "probes" to make this all science. ---
-
-# Carry-over as the important work-in-progress parts of the project here just
-# like above but not as long-standing overarching to the framework but rather
-# for the current hot spots actively being worked on.
-
-# --- START THIS DISCUSSION ---
-
-# Get things started here! Guess at what context should be included.
-# If you get it wrong, you're just wasting 1-turn because the AI will help.
-# Un-comment lines, add lines with absolute-path filenames or `! ` commands.
-
-# Context 1 (Edit-in selections from above and add new files immediately below)
-
-# --- END `adhoc.txt` TEMPLATE ---
+# It is also the portion that mostly comprises the "Prompt Fu" (not FastHTML) side of this system and should always be in `context.txt`.
+
+# --- START `context.txt` TEMPLATE ---
+# context.txt: the list of files an AI will read. context opens it, compile builds it.
+# Keys: j down, k up; Esc then :q quits, :q! discards, :wq saves and quits.
+# One line per thing the AI reads: a file path, or a command after `! `.
+# A line that starts with # is a comment: that file is not read.
+# To add a line: i starts typing, Esc stops. Absolute paths work from anywhere.
+# Web pages, APIs and the connector words: chapter XVIII of foo_files.py.
+
+# --- THE 40K-FOOT VIEW (uncomment on a first turn; comment out on the second) ---
+# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs # <-- the book's spine, one line per article, newest first
+# ~/repos/nixos/autognome.py # <-- the machine's morning routine (this author's NixOS box only)
+# init.lua   # <-- the editor keys that drive the day
+# assets/installer/install.sh # <-- how a stranger's machine gets this workshop
+# GLOSSARY.md   # <-- the terms, defined
+# flake.nix   # <-- the environment, pinned: here is my hardware, here is my state
+# prompt_foo.py   # <-- the compiler that builds the payload
+# foo_files.py   # <-- the router: which files ride, and this book's outline
+# scripts/articles/lsa.py   # <-- the second brain: the article corpus behind `rgx`, `rgxc` and `posts`
+# requirements.in   # <-- the Python packages, pinned
+# pyproject.toml   # <-- the PyPI package
+# __init__.py   # <-- the version

```
+
+# --- ACTUATORS (cheap; include when the AI should be able to act, not only read) ---
+# cli.py                    # <-- tool calls from the command line
+# scripts/xp.py              # <-- turns a pasted reply into the next context
+# scripts/ai.py             # <-- a local AI writes the commit messages
+# scripts/crawl.py           # <-- crawl a site into the next turn
+# scripts/weblogin.py        # <-- warm a login on the persistent browser profile
+# scripts/webclip_2_markdown.py  # <-- a web page, clipped, as markdown
+
+# --- RARE ---
+# scripts/foo_cartridge.py    # <-- the sealed archive: writer and verifier
+# scripts/foo_replay.py       # <-- replay a sealed archive on another machine
+# release.py                 # <-- how a release reaches GitHub and PyPI
+# imports/voice_synthesis.py  # <-- the voice
+# imports/ascii_displays.py   # <-- the ASCII art
+# scripts/release/version_sync.py  # <-- version stamping (to be folded into release.py)
+
+# --- THIS DISCUSSION ---
+# The files and commands for the work in front of you. Paste the NEXT CONTEXT
+# block an AI hands back directly below this line; the AI will correct a guess.
+
+# --- END `context.txt` TEMPLATE ---

 #
================================================================================
====
 # IV. THE WAND🪄 & THE WIZARD🧙 - Everything's actually web development these days
(nix) pipulate $ m
📝 Committing: chore: Update `adhoc.txt` with expanded context template and actuator
definitions
[main d0010c6e] chore: Update `adhoc.txt` with expanded context template and actuator
definitions
 1 file changed, 45 insertions(+), 127 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 48 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1.86 KiB | 317.00 KiB/s, done.
Total 6 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To github.com:pipulate/pipulate.git
   9423da60..d0010c6e  main -> main
(nix) pipulate $
```

And back to the Aider-style exact match interlock patches:
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science.

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
(nix) pipulate $ d
diff --git a/foo_files.py b/foo_files.py
index 8bcb9692..a2704860 100644
--- a/foo_files.py
+++ b/foo_files.py
@@ -1818,14 +1818,14 @@ foo_files.py  # The map + the two forcing-function rules it already
carries.
 """

 ADHOC_CHOP = r"""
-# THE ADHOCPAD (default + a disposable overlay)
-# COMMAND: python prompt_foo.py --chop ADHOC_CHOP --no-tree
-# Add ad hoc files/commands ONLY between the SLOT markers, then wipe them
-# after the campaign. Everything below the slot mirrors DEFAULT_CHOP.
+# THE CONTEXT FILE (context.txt spliced in above a fixed tail)
+# COMMAND: compile   (python prompt_foo.py --chop ADHOC_CHOP --no-tree)
+# context edits context.txt, compile reads it: one line per file or `! `
+# command. Everything below the slot mirrors DEFAULT_CHOP.

 # --- ADHOC SLOT START ---
 # (Structurally empty in the repo. Contents are spliced in at compile
-# time from the gitignored overlay file. Edit with `ahe`. Ordinary git
+# time from context.txt, gitignored. Edit with `context`. Ordinary git
 # staging excludes the overlay, and the pre-commit tripwire refuses it
 # if it is ever forced into the index. For client work, point
 # PIPULATE_ADHOC_FILE outside the worktree so no git sweep can reach it.)
@@ -1837,16 +1837,17 @@ apply.py
 """

 WALK_CHOP = r"""
-# THE WALK ROUTER (only the disposable overlay)
-# COMMAND: PIPULATE_ADHOC_FILE=<router> python prompt_foo.py --frame lean --chop
WALK_CHOP --no-tree --quiet
+# THE LEAN COMPILE (context.txt only, no fixed tail)
+# COMMAND: compile --chop WALK_CHOP --frame lean
 # The frame is a flag beside the chop, not a property of it: the chop picks
 # the files and --frame lean swaps the full checklist for a reading frame.
-# cpr points PIPULATE_ADHOC_FILE at the walk-written router. This chop carries
```

```
+# NO WORD OF ITS OWN SINCE 2026-09-25: cpr and the separate walk router are
+# gone; the walk writes context.txt and compile reads it. This chop carries
 # no fixed tail: every payload line between the slot markers comes from that
-# explicit overlay file.
+# one file, and the flags above reproduce what cpr did.

 # --- ADHOC SLOT START ---
-# (Structurally empty in the repo. cpr supplies the walk router at compile time.)
+# (Structurally empty in the repo. context.txt is spliced in at compile time.)
 # --- ADHOC SLOT END ---
 """

(nix) pipulate $ m
📝 Committing: chore: Update adhoc and walk chop documentation for compile process
[main 98216d52] chore: Update adhoc and walk chop documentation for compile process
 1 file changed, 11 insertions(+), 10 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 48 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 718 bytes | 718.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:pipulate/pipulate.git
   d0010c6e..98216d52  main -> main
(nix) pipulate $
```

This Worm-summoning and this first turn of the discussion ride on Fable 5.1 has been... Impressive.

When you vibe-code and the red-green universal diffs aren't being pinned up in front of you like this I don't think you really appreciate what's going on there in the background, and you certainly can't learn from it and study it and make sure it goes into your git repos with such documentation and audit-friendly transparency, accountability, general web publishing, yadda yadda lots of benefits here and this was a particularly impressive example of...

Well, what the various Opus models I've been trying to use over the past few days would have never been able to do. Wow. Just, wow. Fable 5.1 is not a little bit better than Opus 5 and Opus 5.5. It's on another level.

Ignition?

Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science.

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ exit
exit
(sys) pipulate $ ndq
```

```
(nix) pipulate $ context
(nix) pipulate $
```

Wow, if I read that correct Fable 5.1 has been remarkably thorough in all this.
Okay as part of operation stick bug purging cutsie language we do this:
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science.

```
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ flake
(nix) pipulate $ d
diff --git a/flake.nix b/flake.nix
index 02228964..659c18f4 100644
--- a/flake.nix
+++ b/flake.nix
@@ -1854,7 +1854,7 @@ print(max(1, n))
          # from the top of the screen named which one had run. One body,
          # one first line, typed the same as before.
          g() {
-           clear -x && echo "Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science." && git status
+           clear -x && echo "Left-hand causal boundary \"blast radius\" established. Make 1 change and test." && git status
          }
          m() {
            # THE UNTRACKED-FILE DEBT (banked TODO 2026-07-20, discharged
(nix) pipulate $ exit
exit
(sys) pipulate $ ndq
warning: Git tree '/home/mike/repos/pipulate' is dirty
(nix) pipulate $ g

Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   flake.nix

no changes added to commit (use "git add" and/or "git commit -a")
(nix) pipulate $
```

Okay nice. Oh whoops! If you use the g command then that statement I'm printing:
Left-hand causal boundary "blast radius" established. Make 1 change and test.

...isn't true. If the git repo is dirty under that condition it should read:
Before conducting experiment, type "blast" to establish left-hand causal boundary.
Now let me do the test walk to see the context file at the end of the process. Oh a language change I noticed and want to change. Correct me if I'm wrong about this Fable 5.1 but this makes it feel less redundant and more exact.
Blast Radius Check to establish bisection Left-hand Causal Boundary. It is a Popper-thing. Science.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ rg 'The page is open. When it looks finished, come back to the terminal and type CAPTURE.'
scripts/mother_cat.py
1028:   "The page is open. When it looks finished, come back to the terminal and type CAPTURE.",
(nix) pipulate $ vim scripts/mother_cat.py
(nix) pipulate $ vim scripts/mother_cat.py
(nix) pipulate $ d
diff --git a/scripts/mother_cat.py b/scripts/mother_cat.py
index cec6e723..39f43a75 100644
--- a/scripts/mother_cat.py
+++ b/scripts/mother_cat.py
@@ -1025,7 +1025,7 @@ async def _ride_steps(trail_path, archive, dry_narrate=False, exports_path=None,
         def checkpoint_narration():
             nonlocal disclosed
             disclosed = _narrate(
-                "The page is open. When it looks finished, come back to the terminal and type CAPTURE.",
+   "The page has settled. Come back to the terminal and type CAPTURE.",
                 disclosed,
)

(nix) pipulate $ m
📝 Committing: fix: update narration text in mother_cat.py
[main ba9b0026] fix: update narration text in mother_cat.py
 1 file changed, 1 insertion(+), 1 deletion(-)
(nix) pipulate $ blast
🚀 Pushing 1 commit(s) to remote...
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 48 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 394 bytes | 394.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:pipulate/pipulate.git

```
   8afd51b8..ba9b0026  main -> main

Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ walk
Trail resolved: assets/trails/public_walk.json

This walk opens three public pages. Nothing to sign in to.
Return here and type CAPTURE when prompted at each page.
After all three captures and successful checks, it saves a private summary
and replaces your clipboard. Over SSH it uses a bridge file.
Nothing is sent to a chatbot. Review the summary before sharing it.

Choose a walk:
  1  Practice - read the steps; no pages open.
  2  Start the walk - open the pages.
  q  Exit (Enter also exits).
Choice: 2
Riding trail 'public_walk' -- 3 stop(s).

Make sure your audio is turned on.

This walk opens three pages. You do not need to click anything. At each page, return here and
wait for the CAPTURE prompt. Type CAPTURE and press Enter.
  (If you did not hear that, turn your volume up.)

Summary file: data/decant-preview.md (private; replaced on save).
Checks can miss private details. A blocked check leaves the older file alone.

--- Stop 1/3: the_word ---

Opening page one in the browser. Leave the browser open; it closes on its own after capture.
When the page is ready, return here and type CAPTURE.
Opening the browser; waiting for the page...

The page has settled. Come back to the terminal and type CAPTURE.
Any other response aborts without capturing artifacts.
CAPTURE> CAPTURE
  LOCAL ARCHIVE  /home/mike/repos/pipulate/data/captures/walk-y6vh3_7p/captures.md
(directory 0700, file 0600)
  Captured. final_url=https://npvg.org/walk/1/ artifacts=12
  ADVANCE -> next stop.

--- Stop 2/3: the_receipt ---
```

Opening page two in the browser. Leave the browser open; it closes on its own after capture.
When the page is ready, return here and type CAPTURE.
Opening the browser; waiting for the page...

The page has settled. Come back to the terminal and type CAPTURE.
Any other response aborts without capturing artifacts.
CAPTURE> CAPTURE
  Captured. final_url=https://npvg.org/walk/2/ artifacts=12
  ADVANCE -> next stop.

--- Stop 3/3: the_two_pages ---

Opening the last page in the browser. Leave the browser open; it closes on its own after capture. When the page is ready, return here, type CAPTURE, and read the result.
Opening the browser; waiting for the page...

The page has settled. Come back to the terminal and type CAPTURE.
Any other response aborts without capturing artifacts.
CAPTURE> CAPTURE
  Captured. final_url=https://npvg.org/walk/3/ artifacts=12
  ARCHIVE STATUS complete

Local archive file line for context.txt:
/home/mike/repos/pipulate/data/captures/walk-y6vh3_7p/captures.md
Review locally before compiling; raw bytes are not a safe disclosure.
  CONTEXT FILE NOT UPDATED (ValueError): hand-edited router kept, nothing written; add the line(s) yourself: /home/mike/repos/pipulate/data/captures/walk-y6vh3_7p/captures.md (or delete the file and the next walk rewrites it)
  Archive preserved; the router on disk is unchanged.

Ride complete. Every stop produced a capture receipt.

Checking the summary before saving it and trying the clipboard.
   Preview checks: substitutions=0 denylist=0 secrets=0
   LOCAL PREVIEW /home/mike/repos/pipulate/data/decant-preview.md (0600; sha256=70c289fbd6a171fce23323581a3d378e58509d1c738e8c99d8bf9a69069be82d)
   CONTEXT FILE NOT UPDATED (ValueError): hand-edited router kept, nothing written; add the line(s) yourself: /home/mike/repos/pipulate/data/decant-preview.md /home/mike/repos/pipulate/data/captures/walk-y6vh3_7p/captures.md (or delete the file and the next walk rewrites it)
Markdown output copied to clipboard
   Read the save and copy messages above; either step can fail.
   Review the summary before sharing it. You choose what to send.

The three-page walk is finished. Everything it captured is saved on this computer, and the lines above say whether the summary was saved and copied. Next, type context. That opens a text editor called vim, showing the list of files an AI will read. The keys j and k move the cursor. To leave, press Escape, then type colon q, then Enter. The notes at the top of that file say what to

type after that. Goodbye.

  CAPTURE RUN FINISHED

 Read the save and copy messages above. Either step can fail.
 Review the summary before sharing it.
 Nothing was sent to a chatbot.
---

type one:
  menu   print this list again (useful once it scrolls away)
  walk   take guided tour of context compiler (recommended)
  conn   get mcp, jira, email, docs, etc. into context
  context edit the list of files an AI will read (after the walk)
  prompt   save your clipboard as the question for the AI
  compile build the list and the question into one payload for a chatbot
  all   expanded menu

(nix) pipulate $ context
(nix) pipulate $

Okay wow. The unification has definitely occurred. And it's set to my current job which is good for this instant in time. As we finish this ride we will adjust it so that "Context 1" it gets written by the walk. It's also still called adhoc.txt and I can appreciate why Fable 5.1 didn't jam the rename of that into the prior massive turn. There's scalpel precision here to not as I think I heard it say invalidate past references to adhoc.txt with a giant sed command. We're going to have to measure 3 times and cut once on that one, as if Fable 5.1 isn't already doing that with its sandbox in the background. It is. That's why these patches hardly ever miss anymore. But it's not mixing too many complicated things together which I really appreciate.
Let's plow our way through what's left.
Another thing I noticed is that we have to make these files live in the "assets" folder of the Pipulate repo proper. I went through all the effort of making sure I found stuff that was properly licensed to replace jeopardy.wav so we should incorporate them properly. I noticed the silence when I was on the Mac.
(nix) pipulate $ pwd
/home/mike/.local/share/pipulate
(nix) pipulate $ ls
ding.wav tick.wav
(nix) pipulate $

Oh and to make the language more consistent with my recent change:
(nix) pipulate $ vim scripts/mother_cat.py
(nix) pipulate $ rg 'When the page is ready'
assets/trails/public_walk.json
19:      "guidance": "Opening page one in the browser. Leave the browser open; it closes on its own after capture. When the page is ready, return here and type CAPTURE.",
32:      "guidance": "Opening page two in the browser. Leave the browser open; it closes on its own after capture. When the page is ready, return here and type CAPTURE.",

```
45:      "guidance": "Opening the last page in the browser. Leave the browser open; it closes on its own after capture. When the page is ready, return here, type CAPTURE, and read the result.",
(nix) pipulate $ vim assets/trails/public_walk.json
(nix) pipulate $ d
diff --git a/assets/trails/public_walk.json b/assets/trails/public_walk.json
index 6bb5ba4b..ef5379a0 100644
--- a/assets/trails/public_walk.json
+++ b/assets/trails/public_walk.json
@@ -16,7 +16,7 @@
     {
       "name": "the_word",
       "label": "The capture word",
-      "guidance": "Opening page one in the browser. Leave the browser open; it closes on its own after capture. When the page is ready, return here and type CAPTURE.",
+      "guidance": "Opening page one in the browser. Leave the browser open; it closes on its own after capture. When the page has settled, return here and type CAPTURE.",
       "url": "https://npvg.org/walk/1/",
       "target_slot": "slot_one",
       "harvest_regex": ".+",
@@ -29,7 +29,7 @@
     {
       "name": "the_receipt",
       "label": "Page two",
-      "guidance": "Opening page two in the browser. Leave the browser open; it closes on its own after capture. When the page is ready, return here and type CAPTURE.",
+      "guidance": "Opening page two in the browser. Leave the browser open; it closes on its own after capture. When the page has settled, return here and type CAPTURE.",
       "url": "https://npvg.org/walk/2/",
       "target_slot": "slot_two",
       "harvest_regex": ".+",
@@ -42,7 +42,7 @@
     {
       "name": "the_two_pages",
       "label": "Two versions of one page",
-      "guidance": "Opening the last page in the browser. Leave the browser open; it closes on its own after capture. When the page is ready, return here, type CAPTURE, and read the result.",
+      "guidance": "Opening the last page in the browser. Leave the browser open; it closes on its own after capture. When the page has settled, return here, type CAPTURE, and read the result.",
       "url": "https://npvg.org/walk/3/",
       "target_slot": "slot_three",
       "harvest_regex": ".+",
(nix) pipulate $ m
📝 Committing: chore: Refine guidance text in public_walk.json
[main 7cb51ba2] chore: Refine guidance text in public_walk.json
 1 file changed, 3 insertions(+), 3 deletions(-)
(nix) pipulate $ git push
```

Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 48 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 477 bytes | 477.00 KiB/s, done.
Total 5 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To github.com:pipulate/pipulate.git
   ba9b0026..7cb51ba2 main -> main
(nix) pipulate $

Okay wow there. I feel pretty good about this.
4: Prompt: THE AFTER READING for the context.txt ride, one probe at a time, deltas over absolutes:

1. The adhoc census (rg -c by file): prompt_foo.py -7, apply.py -3, scripts/mother_cat.py -4, init.lua -6, flake.nix -8 (three patched lines plus the five inside the deleted span), foo_files.py -4 (two markers, the ASCII art line, the chapter III lead-in); .gitignore holds at 2 by design. Every other file's count is unchanged and is next turn's list.
2. The flake definitions: four lines before (context, _walkrouter, epr, cpr), one after (context), and the context line now names the editor fallback and context.txt.
3. boot_menu --recall: the epr and cpr rows are gone; context, prompt and compile rows sit between conn and all; seven rows.
4. mother_cat.py: three hits before (with_name, REPO_ROOT adhoc.txt, "type epr"), two after (REPO_ROOT context.txt, "type context").
5. init.lua patterns: two lines before, one after (context.txt).
6. template_lines: read the old number (about 110), 43 after.
7. check-ignore: NOT_IGNORED before; .gitignore:LINE:context.txt after.
8. The PIPULATE_ADHOC_FILE probe is a CENSUS and reads the same in both taps; it names the file that exports the variable, and that file is the target of this turn's rename car.
9. The nvim gate prints "init.lua loaded successfully!" in both taps; it is a gate, not a landing. Car receipts to confirm from the terminal pastes: INIT_LUA_MERGED (24 old, 18 new), WALK_ROUTER_WORDS_REMOVED (67 lines; more means the span carried empty lines), NIX_PARSE_OK, TEMPLATE_REPLACED, and the Paintbox line reading +0 claimed.

THIS TURN'S CARS, once the AFTER is in: (a) The rename on this machine: mv ~/.local/state/pipulate/adhoc.txt to context.txt, rm ~/.local/state/pipulate/adhocwalk.txt, and change the export named by probe 8 to the new path, in one gated paste; then prune the live file's header down to the plain template with a python car that keeps every non-comment line (gate: the file's first line still starts with "# AD HOC CHOP!" and there is exactly one "# Context 1 (Edit-in" line; if the header is already gone by hand, the gate says so and writes nothing). (b) The census leftovers by file: scripts/xp.py (does it write to adhoc.txt), scripts/git_hooks/pre-commit (the index tripwire must refuse context.txt), introduction.md, scripts/walk.py's docstring, scripts/sources_menu.py's docstring ("type sources"), and the § keys in foo_files.py that say adhoc.txt (PROBE ECHO RULE and its siblings; never the dated receipts, which are history). (c) Rulings owed, not cars: whether the reading frame gets a word back or stays compile --frame lean; whether PIPULATE_ADHOC_FILE, ADHOC_CHOP and the ADHOC SLOT markers are renamed (each is an env var or a regex with readers in three files, so each is its own gated car with the readers in context); whether page 3 of the walk should

name prompt and compile beside its suggested question (a nixops.sh truck, not a git push). [The prompt you want AFTER experiment conducted and reading is in; probably what the AI gives you, but think for yourself!]
5: Deliverables: A much easier system to describe and fewer and more obvious commands to remember.
Note: I'm going to get ahead of Fable 5.1 thinking here a bit as interesting as it is to watch. I have to say that new sentence per what would have been a paragraph before is actually quite nice. The sentences are good signal-to-noise ratio decisions and I'm sure a balance with token generation and level of exposure of actual internal and potentially proprietary Anthropic framework stuff. It's growing on me and I like to copy-paste the stacked sentences as much as I did that more granular full paragraph-style conversational HTML from the past. I can see the surgical measure three times cut once work going on. Renaming adhoc.txt to context.txt is a perilous move but necessary.
Another thing I'm going to do is to rework the ASCII bunny for operation stick bug. Here's what's there currently. I've stripped out the colorizing for now.

```
                ( Like a [[[canary]]] you say? )
                                  O       /) ____          The "No Problem" Framework
>  I HEREBY WILL NOT RE-GENERATE          o /)\__//  /    \       Pipulate - Protecting Your
Code
>  Once upon machines be smarten          ___(/_ 0 0  |     |      just by being honest about text.
>  ASCII sealing immutata art in       *(    ==(_T_)== NPvg |       (If mangled, then AI drifted.)
>  This here cony if it's broken       \  )   ""\  |     |           https://pipulate.com
>  Smokin gun drift now in token         |__>-\_>_>  \____/                  🥕🥕🥕
```

This is too weird and off the beaten track. We need it to make them feel the Eric Raymond Cathedral in the bizarre style itch. Some of the points we want to get in the rework are:
Because Claude said so... that's a promissory note or the equivalent of a Cockpit Voice Recorder (CVR and not as good as a Flight Data Recorder (FDR)
Are you forwarding that Claude output to as client? Are you using what it said to change a site in production? Can you put your name behind that? Is that a confident hallucination? A confabulation? How do you know?
We are not really in the certification game yet but we do want to imply how serious it is to not be able to reproduce "what Claude did" in the background. If you can't reproduce it yourself, you cannot assure it's quality. Something like that. Maybe some W. Edwards Deming TQM inspired language. Here's certifications that may be relevant.

Framework	Where	What it asks for	What a turn already produces
NIST SP 800-53 Rev. 5	CM-2, CM-3, CM-4, SA-10, SI-7	baseline, change control, impact analysis, developer CM, integrity	commit as baseline, one diff, before/after, digest
NIST SP 800-218 (SSDF)	PS.3, PW.8	archive and provenance; test executable code	cartridge, straddle
ISO/IEC 27001:2022	A.8.9, A.8.32	configuration and change management	lock file, the train

Framework	Where	What it asks for	What a turn already produces
SOC 2 (AICPA TSC)	CC8.1	authorize, design, test, approve, implement	patch, app, d, m
PCI DSS v4.0	6.5.1, 6.5.2	change control; confirm changes worked	AFTER against BEFORE
ITIL 4	change enablement, standard change	pre-authorized, low-risk, repeatable	one car, one commit
COBIT 2019	BAI06, BAI10	managed changes, managed configuration	same
ISO 9001:2015	8.5.6, 10.2	control of changes; corrective action with root cause	bisection to the smoking gun
SLSA	build provenance	who built what from what	manifest.json, digest
SPDX / CycloneDX, EO 14028	SBOM	dependency inventory	flake.lock
Reproducible Builds	bit-for-bit	same input, same bytes	Nix store, byte-reproducible cartridge
CIS Controls v8	4, 8, 16	secure configuration, audit logs, application security	declared environment, receipts
DO-178C, IEC 61508, ISO 26262, IEC 62304	traceability, verification, change control	(the HRO vocabulary)	receipts trace change to evidence
21 CFR Part 11	audit trails	tamper-evident records	hash-sealed archives

We also want to get us to go to this very soon:

```
Workshop/
├── corporate/              lowerdir — read-only, replaced wholesale
│   ├── AGENTS.md             org canon
│   ├── .agents/skills/       org skills
│   └── log.md              ◄── the history of what's been made official.
│
├── shared/
│   ├── mike/               ONE writer. Ever.
│   │   ├── .identity         the claim. collides here or nowhere.
│   │   ├── AGENTS.md         MY conventions — nearest-ancestor wins
│   │   │                     INSIDE my namespace, invisible outside it
│   │   └── .agents/skills/   my skills, overlaid on corporate's
│   └── alice/              ONE writer. Ever. Never me.
└── personal/ ≡ Playground/   gitignored. Structurally unreachable
                              by any git verb. rm is the only threat.
```

And what's more we anticipate people auditing the Pipulate project with the intention of shutting it down and forbidding the LLM Q/A work by virtue of the scary and unfamiliar things it calls for like installing Nix with the DetSys installer. We plan on answering them preemptively with one or

more of the following techniques which we can start putting to work immediately with the ASCII bunny reworking.

Thirty moves for a review that arrives wanting to shut the thing down, then three.

1. Answer the checklist before it is read aloud: the repository's own briefing takes each audit question and names the artifact that answers it.
2. Ask the reciprocal question, with courtesy: what record does the alternative produce, and who can verify it with the author absent?
3. Hand over the instrument, not the argument: the archive, the verify command, nothing to install.
4. Speak in control identifiers, never in tool names: CM-3 before Nix, provenance before cartridge.
5. Grant the premise entirely: unvetted AI-generated code is a hazard; this is the vetting.
6. Reframe the object: not an unsanctioned tool but the QA instrument for sanctioned ones.
7. Use their smallest category on purpose: a standard change, pre-authorized, low-risk, repeatable.
8. Shrink the footprint until nothing is left to object to: one directory tree, no daemons, no listeners beyond localhost.
9. Make the record outlive the maker: a reviewer must never need you in the room.
10. Publish the digest beside the archive: tamper-evidence costs one line and replaces trust.
11. Request their control catalogue and map onto it line by line, in writing.
12. Volunteer for the audit before it is scheduled: the reluctant are audited, the willing are consulted.
13. Show the tripwire firing: a payload blocked for a credential shape is the demonstration nobody expects.
14. Time-box the meeting: one page, nine questions, ten minutes, then the archive.
15. Cite bisection by its git name: opposing git bisect is opposing a built-in the organization already ships.
16. Separate the two fears and address only one: loss of control has a remedy; loss of relevance has none you can offer.
17. Keep a paper trail of courtesy: every exchange in writing, every claim beside its receipt.
18. Invite the red team: a fence observed only refusing is unwitnessed, and their attempt is your receipt.
19. Distinguish policy from tool in the first sentence: policy governs where data goes; this governs how change is recorded.
20. Never oppose a certification; adopt its vocabulary and let the gap speak for itself.
21. Escalate by evidence, never by rank: the archive travels to whoever asks.
22. Name the asymmetry plainly: the unrecorded change is the one no audit can reach.
23. Offer the procedure for their own changes: a toggle flipped in a console is a change too, and it has no diff.
24. Make a clean laptop the control group: every claim reproducible from the lock file alone.
25. Be boring on purpose: no personas, no demos, only diffs and digests.
26. Name the risk they already own: AI-assisted work is inside the perimeter now, recorded or not.
27. When refused, ask which property they would like removed, and write the answer down.
28. Keep the client's data out of the argument structurally: gitignore, denylist, tripwire, and the scrub line on every compile.
29. Let the vocabulary do the flattering: provenance, baseline, reproducible are their words, spoken back.

30. Keep working: the record accrues on every turn regardless, and the record is the argument.

The three. Numbers 1 and 2 together, the Socratic pair: answer every question before it is asked, then put the same question to everything else in the estate. Load-bearing because it moves the review from opinion about tools to evidence about records, and because the reciprocal question has no answer for unrecorded work, so the reviewer reaches the corner unassisted. Car 2 below writes exactly this into AI_CONTEXT.md. Numbers 3 and 23, the instrument offered: hand them the verify command and propose the procedure for their own changes. Load-bearing because it dissolves the adversary; a reviewer who has run the verifier has used the tool, and a control that then forbids it forbids its own audit. Numbers 27 and 30, the written refusal and the accruing record: ask which property they would remove, record the answer, keep working. Load-bearing because it is the mechanism of the sentence above: a refusal becomes one more receipt rather than an outcome, and time does the arguing.

Okay, now back to our regularly scheduled Fable 5.1 response. Here's 12 minutes and 26 seconds worth of thought:

Reviewing live command receipts to verify the current state.

Comparing per-file match counts between before and after receipts.

Tracing displacement shifts across router and menu files.

Reconciling line-count deltas against the diff statistics.

Searching configuration files for the missing environment variable source.

Widening the search to nested configuration files for the variable.

Cataloging remaining outdated file references across the codebase.

Counting scattered file-name references across documentation and scripts.

Tallying remaining file-name mentions across scattered receipts.

Locating the actual export before pairing rename with variable change.

Expanding the search to shell profiles and repo config files.

Weighing whether to bundle the rename with the variable fix.

Designing the header replacement while preserving structured sections.

Deciding which files the next compile actually needs.

Locating the marker line that anchors the discussion section.

Deciding whether to keep a backup copy of the file.

Auditing scattered files for outdated adhoc-related references.

Extending the tripwire loop to cover the renamed overlay file.

Rewriting newcomer-facing instructions to match renamed commands.

Updating stale docstrings to match the renamed command names.

Rewriting the handoff paragraph to drop stale command names.

Rewriting the handoff block to reference the new filename.

Organizing remaining fixes into separate justified commit-sized batches.

Deferring audio-file relocation pending missing loader-script context.

Proposing an append-only merge strategy for hand-edited files.

Drafting conditional logic for the git status banner function.

Locating stale adhoc.txt comments to rename across the config.

Applying the same rename fix across editor configuration files.

Rewriting scattered file references to match the new filename.

Deciding whether an outdated todo now warrants deletion.

Tallying which scattered file references still need updating.

Rewriting the handoff paragraph to remove obsolete alias references.

Choosing the more accurate wording for the load-event message.

Bundling newcomer-facing string fixes into one commit.

Designing the safe search for the environment variable's file path.

Writing the header-pruning script to rebuild the context file.

Verifying line counts in the extracted header template.

Determining which files the next context turn requires.

Listing files needed for the next round of patches.

Refining probe commands to avoid false-zero readings.

Counting adhoc references across files before and after fixes.

Designing a robust probe to confirm the shell-hook text.

Finding lingering mentions of a retired term in comments.

Rewriting stale mechanism comments describing a retired command term.

Rewriting cpr comments to describe the current flag mechanism.

Extending the outdated-term census to more repository files.

Predicting file-count deltas after removing outdated term mentions.

Rewriting a comment to remove a lingering outdated term.

Rewriting error and status messages to match the renamed file.

Renaming lingering references to the mastery-toggle keybinding.

Rewriting the git status banner to add conditional checks.

Updating documentation comments to replace an outdated scratchpad filename.

Rewriting outdated command names across explanatory comment blocks.

Sequencing the remaining file patches into separate commits.

Weighing whether stale context should be pruned or left riding.

Drafting the header-pruning script to comment out stale lines.

Extracting the reusable template ahead of the header rewrite.

Drafting the next context file's file list and probes.

Checking shell quoting safety across the probe command list.

Weighing whether the sound files belong in version control.

Designing an append-only guard so walks never overwrite edits.

Rewriting newcomer instructions to match renamed commands and files.

Adjusting frame-selection comments to match the current default flag.

Cataloging vanished filenames caused by zero-match omissions.

Counting mechanism comments still naming the retired term.

Extracting exact comment lines for the renamed-term batch.

Predicting file-count changes after the remaining renames land.

Rewriting the handoff comment block for the renamed overlay file.

Rewriting the disappearing-instrument passage to drop retired terms.

Drafting an honest reply about the process actually used.

Locating the pre-commit block that guards the overlay file.

Running verification probes before finalizing the remaining patches.

Verifying probe commands for quoting and hidden-search accuracy.

Drafting the next context handoff and remaining fix summary.

Extending the outdated-term census to installer and launcher scripts.

Verifying exact comment substrings before drafting the file rewrite.

Drafting the final patch cars and terminal instructions.

Fable 5.1 Extra: Context verified. Every file a car below patches is in this payload: scripts/git_hooks/pre-commit, introduction.md, scripts/walk.py, scripts/sources_menu.py, scripts/mother_cat.py, init.lua, prompt_foo.py, flake.nix, foo_files.py. Two files next turn needs are absent and named in NEXT CONTEXT: tools/scraper_tools.py (the sound loader) and whatever file exports PIPULATE_ADHOC_FILE, which this compile failed to find.

Two corrections before the reading

No sandbox. I answer from the payload's raw source, and the patches land because the interlock refuses anything that is not byte-exact; the two misses below are the places I counted instead of measured. The pinned-up diffs are the thing worth being impressed by, and they are yours: without them a rename like this is a vibe with no receipt.
On "settled": the guidance in public_walk.json is spoken before the page opens and puts the judgment on the human, so "when the page has settled, return here" is fine. The checkpoint narration is different: it fires when driver.get returns, which is the load event, and the guided lane skips the twenty-second settle wait (the 2026-09-19 receipt read this off the scraper). So "the page has settled" is a verb no code performed; "the page has loaded" is exactly what happened. Car 5 makes that one-word swap and leaves your JSON alone.

The AFTER reading

1. Census, deltas exact on all six: prompt_foo.py 14→7, apply.py 3→0, mother_cat.py 4→0,

init.lua 8→2, flake.nix 14→6, foo_files.py 22→18; .gitignore holds at 2. rg -c prints nothing for a file at zero, so apply.py and mother_cat.py vanishing from the list is the zero.

2. Flake definitions 4→1; context() moved 1526→1531, the five comment lines Car 7 put above it.
3. Recall prints seven rows; epr and cpr are gone.
4. mother_cat.py 3→2, both survivors displaced +4 (ROUTER_TAIL grew by four).
5. init.lua 2→1 at line 65; 72−7 = 65, since 25 old lines became 18.
6. Template 125→43. The diffstat's +45/−127 is the block count plus the lead-in and one blank git chose to swap; the net −82 is invariant.
7. .gitignore:47:context.txt.
8. The export probe is the finding: every candidate file was empty and ENV= still reads the path. A census with no member is an instrument failure, not an empty world; the enumeration was wrong, so the rename car waits on a wider one.
9. The nvim gate is green on both taps, as a gate should be.

Two misses, both mine: INIT_LUA_MERGED read 25 old against 24 predicted (a nine-line comment counted as eight), and the template read 125 against "about 110." WALK_ROUTER_WORDS_REMOVED 67, NIX_PARSE_OK, TEMPLATE_REPLACED and the Paintbox +0 all landed.

The walk's two refusals are the unification's witness: both writes aimed at the file context opens, found 32 lines of yours, printed their two paths and touched nothing. The guard exists so a walk never overwrites a human's list. Making the walk write a "Context 1" block is a design change, not a rename: the honest shape is an append under a dated marker, never a replace, and it needs mother_cat.py in a turn of its own.

1. PROBES

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter.

```
rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate ~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc /etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
for f in scripts/git_hooks/pre-commit .git/hooks/pre-commit; do echo "$f:$(rg -c 'context\.txt' "$f" || echo 0)"; done
rg -cw 'epr|cpr' --sort path prompt_foo.py foo_files.py flake.nix init.lua introduction.md GLOSSARY.md scripts assets/installer remotes/honeybot/www/npvg.org
for f in scripts/walk.py init.lua scripts/mother_cat.py; do echo "$f:$(rg -c adhoc "$f" || echo 0)"; done
rg -n 'door 2' scripts/sources_menu.py || echo door2=0
LD_LIBRARY_PATH="" nix eval --raw .#devShells.x86_64-linux.quiet.shellHook 2>/dev/null | rg -c 'Before conducting experiment' || echo 0
nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
ls -la ~/.local/share/pipulate/; rg -n 'local/share/pipulate|tick\.wav|ding\.wav|\*\.wav' tools/scraper_tools.py .gitignore
```

Line 1 is the rename census. Line 2 widens the export hunt to the repo's own .env (the flake sources it) and NixOS's generated files; it is the ruling for next turn's rename car. Line 3

straddles the hook twice: the scripts/ copy moves at patch, the .git/hooks copy moves at ignition, so a reading of N and 0 says the shell was not re-entered. Line 4 is the retired-word census; scripts/boot_menu.py will read 1 forever, the epitaph in the comment I wrote. Line 5 reads walk.py 2→1 (its JSON key adhoc_mutation stays until the ADHOC-name ruling), init.lua 2→0, mother_cat.py 1→0. Line 6 is 2→0. Line 7 reads the generated hook text, the only witness of a shell function. Line 8 is a gate. Line 9 is the CENSUS for the sound car.

2. NEXT CONTEXT

```
# --- NEXT TURN'S CARS: the rename (once line 2 names the export), the sounds, the walk's write ---
scripts/mother_cat.py
tools/scraper_tools.py
.gitignore
# flake.nix        # <-- only if Car 8 or 9 refused; line 7 is their witness
# foo_files.py     # <-- only if Car 10 refused, or to rule on the six dated adhoc lines it keeps
# init.lua         # <-- only if Car 6 refused
# introduction.md  # <-- only if Car 3 refused
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate ~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc /etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
! for f in scripts/git_hooks/pre-commit .git/hooks/pre-commit; do echo "$f:$(rg -c 'context\.txt' "$f" || echo 0)"; done
! rg -cw 'epr|cpr' --sort path prompt_foo.py foo_files.py flake.nix init.lua introduction.md GLOSSARY.md scripts assets/installer remotes/honeybot/www/npvg.org
! for f in scripts/walk.py init.lua scripts/mother_cat.py; do echo "$f:$(rg -c adhoc "$f" || echo 0)"; done
! rg -n 'door 2' scripts/sources_menu.py || echo door2=0
! LD_LIBRARY_PATH="" nix eval --raw .#devShells.x86_64-linux.quiet.shellHook 2>/dev/null | rg -c 'Before conducting experiment' || echo 0
! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
! ls -la ~/.local/share/pipulate/; rg -n 'local/share/pipulate|tick\.wav|ding\.wav|\*\.wav' tools/scraper_tools.py .gitignore
```

3. PATCHES

Ten cars. Car 1 is a terminal paste that changes a file outside the repo, so it has no d and no m; its receipt is the line it prints. Cars 2 through 10 are patches: patch, app, d, m each.

Car 1 (terminal paste): the live file's header becomes the plain template

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter. It reads the template out of foo_files.py (one authority), keeps every line below your #

Context 1 marker, and turns the lines that were active into comments so the next compile carries only what you paste from NEXT CONTEXT. Nothing is deleted. The file keeps its old name until line 2 finds the export.

```
python - <<'EOF'
import os, sys
from pathlib import Path
root = Path(os.environ.get("PIPULATE_ROOT", "."))
sel = os.environ.get("PIPULATE_ADHOC_FILE", "")
p = Path(sel).expanduser() if sel.strip() else root / "context.txt"
if not p.is_file():
    sys.exit(f"STOP: no context file at {p}")
lines = p.read_text(encoding="utf-8").split("\n")
if not lines[0].startswith("# AD HOC CHOP"):
    print(f"HEADER_ALREADY_PRUNED {p} first_line={lines[0][:50]!r}; nothing written")
    sys.exit(0)
C = [i for i, l in enumerate(lines) if l.startswith("# Context 1 (Edit-in")]
assert len(C) == 1, ("STOP: Context 1 marker", C)
ff = (root / "foo_files.py").read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(ff) if l.rstrip() == "# --- START `context.txt` TEMPLATE ---"]
E = [i for i, l in enumerate(ff) if l.rstrip() == "# --- END `context.txt` TEMPLATE ---"]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: template markers", S, E)
template = ff[S[0] + 1:E[0]]
while template and not template[-1].strip():
    template.pop()
tail, muted = [], 0
for l in lines[C[0] + 1:]:
    if l.strip() and not l.lstrip().startswith("#"):
        l, muted = "# " + l, muted + 1
    tail.append(l)
out = template + ["", "# Context 1"] + tail
p.write_text("\n".join(out), encoding="utf-8")
print(f"HEADER_PRUNED {p} header_lines_replaced={C[0] + 1} template={len(template)}
tail_kept={len(tail)} tail_muted={muted}")
EOF
```

Car 2 (patch): the commit airlock refuses all three names

```
Target: scripts/git_hooks/pre-commit
[[[SEARCH]]]
if git ls-files --error-unmatch -- adhoc.txt >/dev/null 2>&1; then
    echo "🛑 COMMIT BLOCKED: adhoc.txt is tracked in the index."
    echo "   The adhoc overlay must never enter history."
    echo "   Fix: git restore --staged adhoc.txt   (or git rm --cached adhoc.txt)"
    exit 1
fi
[[[DIVIDER]]]
for overlay in context.txt adhoc.txt adhocwalk.txt; do
```

```
    if git ls-files --error-unmatch -- "$overlay" >/dev/null 2>&1; then
        echo "🛑 COMMIT BLOCKED: $overlay is tracked in the index."
        echo "   The context file (and its two retired names) must never enter history."
        echo "   Fix: git restore --staged $overlay   (or git rm --cached $overlay)"
        exit 1
    fi
done
[[[REPLACE]]]
Target: scripts/git_hooks/pre-commit
[[[SEARCH]]]
    echo "   gitignored overlay (edit with ahe)."
[[[DIVIDER]]]
    echo "   gitignored context file (edit with context)."
[[[REPLACE]]]
```

Car 3 (patch): introduction.md teaches the three words

```
Target: introduction.md
[[[SEARCH]]]
They edit a plain text file called the prompt router by typing epr. Every
line in it that does not start with # is one thing you will see the next
time they compile: a file path, or a shell command written after `! ` whose
output rides along. They compile it by typing cpr and paste the result here
again. The full archive, captures.md, is named in that file but commented
out, because it is unreviewed and may hold private details.
[[[DIVIDER]]]
They edit a plain text file called context.txt by typing context. Every
line in it that does not start with # is one thing you will see the next
time they compile: a file path, or a shell command written after `! ` whose
output rides along. They save a question from their clipboard by typing
prompt, build the file and the question into one payload by typing compile,
and paste the result here again. The full archive, captures.md, is named in
that file but commented out, because it is unreviewed and may hold private
details.
[[[REPLACE]]]
Target: introduction.md
[[[SEARCH]]]
say so plainly and name where it would be found: a lens in the full archive,
a fresh walk, or a router line. When you suggest adding something to the
router, give the exact line to type, and say whether the person should read
that file themselves before compiling it. Use only the words they have
already met at the terminal (menu, walk, epr, cpr) and in the editor (j, k,
Esc, :q, :q!, :wq). Do not ask them to install anything.
[[[DIVIDER]]]
say so plainly and name where it would be found: a lens in the full archive,
a fresh walk, or a line in context.txt. When you suggest adding something to
context.txt, give the exact line to type, and say whether the person should
```

read that file themselves before compiling it. Use only the words they have already met at the terminal (menu, walk, context, prompt, compile) and in the editor (j, k, Esc, :q, :q!, :wq). Do not ask them to install anything.
[[[REPLACE]]]

Car 4 (patch): two docstrings stop naming retired words

Target: scripts/walk.py
[[[SEARCH]]]
browser, voice, shell, or adhoc.txt mutation path in this file.
[[[DIVIDER]]]
browser, voice, shell, or context.txt mutation path in this file.
[[[REPLACE]]]
Target: scripts/sources_menu.py
[[[SEARCH]]]
what door 2 opens onto.
[[[DIVIDER]]]
what `conn` prints.
[[[REPLACE]]]
Target: scripts/sources_menu.py
[[[SEARCH]]]
boot_menu.py's door 2 drops the human into the Nix shell and tells them to type `sources`. This is the roster they get: the commands that reach OUTSIDE
[[[DIVIDER]]]
boot_menu.py lists `conn` at every shell entry and the Nix shell defines it. This is the roster it prints: the commands that reach OUTSIDE
[[[REPLACE]]]

Car 5 (patch): the strings a newcomer read on today's walk

Target: scripts/mother_cat.py
[[[SEARCH]]]
            "The page has settled. Come back to the terminal and type CAPTURE.",
[[[DIVIDER]]]
            "The page has loaded. Come back to the terminal and type CAPTURE.",
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
            "hand-edited router kept, nothing written; add the line(s) yourself: "
[[[DIVIDER]]]
            f"{target.name} has lines of your own, so nothing was written; add the line(s) yourself:
"
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
        print(" Archive preserved; the router on disk is unchanged.")

[[[DIVIDER]]]
        print(" Archive preserved; the context file on disk is unchanged.")
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
        prefix=".adhocwalk-", delete=False,
[[[DIVIDER]]]
        prefix=".context-", delete=False,
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
THE TRAINING WHEELS (2026-09-20, read off the first Mac epr screen). `epr`
[[[DIVIDER]]]
THE TRAINING WHEELS (2026-09-20, read off the first Mac epr screen). `context`
[[[REPLACE]]]

Car 6 (patch): init.lua's two leftovers

Target: init.lua
[[[SEARCH]]]
   -- the column) and tw=0 (long lines run free for adhoc.txt masthead art
[[[DIVIDER]]]
   -- the column) and tw=0 (long lines run free for context.txt lines
[[[REPLACE]]]
Target: init.lua
[[[SEARCH]]]
      .. "4. SEED: the adhoc.txt lines (and TODO_SLUGS if narrative context is\n"
[[[DIVIDER]]]
      .. "4. SEED: the context.txt lines (and TODO_SLUGS if narrative context is\n"
[[[REPLACE]]]

Car 7 (patch): the compiler's own comments stop naming cpr and adhoc.txt

Target: prompt_foo.py
[[[SEARCH]]]
      # ahead of the prompt. "lean" is the reading frame cpr passes, for a
      # question asked of a walk preview. Nothing reads the prompt text to
[[[DIVIDER]]]
      # ahead of the prompt. "lean" is the reading frame for a question asked
      # of a walk preview (compile --frame lean). Nothing reads the prompt text to
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
      and the train do not, because nothing in a cpr compile is asking for

[[[DIVIDER]]]
      and the train do not, because nothing in a lean compile is asking for
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
   # selects files; a frame selects what rides ahead of the prompt. cpr in
   # flake.nix passes --frame lean; every other alias omits it and gets the
   # default, so no existing compile changes shape. The TODO that seeded this
[[[DIVIDER]]]
   # selects files; a frame selects what rides ahead of the prompt. Since
   # 2026-09-25 no alias passes it: compile --frame lean is the spelling, and
   # every compile keeps the default until a hand types it. The TODO that seeded this
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
                # Marker parity: the payload label IS the adhoc.txt line.
[[[DIVIDER]]]
                # Marker parity: the payload label IS the context.txt line.
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
   adhoc.txt therefore witnesses THIS compile's fill, not the previous one --
[[[DIVIDER]]]
   context.txt therefore witnesses THIS compile's fill, not the previous one --
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
   BEFORE the `!` executor loop -- so a probe echoed into adhoc.txt witnesses
[[[DIVIDER]]]
   BEFORE the `!` executor loop -- so a probe echoed into context.txt witnesses
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
   # a `! ... prompt_foo.py ...` line in adhoc.txt makes the compiler run a
[[[DIVIDER]]]
   # a `! ... prompt_foo.py ...` line in context.txt makes the compiler run a
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
splices the same adhoc.txt
[[[DIVIDER]]]
splices the same context.txt
[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
      # echoes lost their leading "! " during a hand-copy into adhoc.txt, so
[[[DIVIDER]]]
      # echoes lost their leading "! " during a hand-copy into context.txt, so

[[[REPLACE]]]
Target: prompt_foo.py
[[[SEARCH]]]
        # filename cited mid-thought (index.md, SKILL.md, adhoc.txt), or a
[[[DIVIDER]]]
        # filename cited mid-thought (index.md, SKILL.md, context.txt), or a
[[[REPLACE]]]

Car 8 (patch): flake.nix's six comment lines

Target: flake.nix
[[[SEARCH]]]
    # adhoc.txt inherit them through PATH just like interactive Bash does.
[[[DIVIDER]]]
    # context.txt inherit them through PATH just like interactive Bash does.
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
    # chisel-strike in adhoc.txt spawns a non-interactive child that
[[[DIVIDER]]]
    # chisel-strike in context.txt spawns a non-interactive child that
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
adhoc.txt `!` lines
[[[DIVIDER]]]
context.txt `!` lines
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
        # Lines ACCUMULATE on purpose. adhoc.txt is a scratchpad, `ahe` is
[[[DIVIDER]]]
        # Lines ACCUMULATE on purpose. context.txt is a scratchpad, `context` is
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
        # adhoc.txt. The only honest witness is a human pressing Tab.
[[[DIVIDER]]]
        # context.txt. The only honest witness is a human pressing Tab.
[[[REPLACE]]]
Target: flake.nix
[[[SEARCH]]]
        # functions, so interactive use and adhoc.txt child shells share one implementation.
[[[DIVIDER]]]
        # functions, so interactive use and context.txt child shells share one implementation.
[[[REPLACE]]]

Car 9 (patch): g prints a verdict, not a slogan

Target: flake.nix
[[[SEARCH]]]
```
        # one first line, typed the same as before.
        g() {
          clear -x && echo "Left-hand causal boundary \"blast radius\" established. Make 1 change
and test." && git status
        }
```
[[[DIVIDER]]]
```
        # one first line, typed the same as before.
        # THE FIRST LINE IS A VERDICT (2026-09-25, operator's ruling): the
        # boundary is a commit, so a dirty tree has none yet and the line
        # says what makes one; a clean tree has it and the line says so.
        g() {
          clear -x
          if [ -n "$(git status --porcelain)" ]; then
            echo "Before conducting experiment, type \"blast\" to establish left-hand causal
boundary."
          else
            echo "Left-hand causal boundary \"blast radius\" established. Make 1 change and test."
          fi
          git status
        }
```
[[[REPLACE]]]

Car 10 (patch): the router's rules and chapter VIII-b name the one file

Target: foo_files.py
[[[SEARCH]]]
line into the next adhoc.txt; the hand-run is BEFORE
[[[DIVIDER]]]
line into the next context.txt; the hand-run is BEFORE
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
adhoc.txt lines. Mechanism: init.lua's <leader>k stages the canned block
[[[DIVIDER]]]
context.txt lines. Mechanism: init.lua's <leader>k stages the canned block
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
(adhoc.txt is this book's margin)
[[[DIVIDER]]]
(context.txt is this book's margin)

[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
the adhoc.txt pasted into an article and the adhoc.txt the compiler READ
[[[DIVIDER]]]
the context.txt pasted into an article and the context.txt the compiler READ
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
adhoc.txt then reads its continuation lines
[[[DIVIDER]]]
context.txt then reads its continuation lines
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
DECANT, the adhocwalk.txt writer
[[[DIVIDER]]]
DECANT, the context.txt writer
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
Compiles the selected router (adhocwalk.txt: `epr` edits it, `cpr` compiles it) into a payload
[[[DIVIDER]]]
Compiles context.txt (`context` opens it, `compile` builds it) into a payload
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
Selection: adhocwalk.txt names the checked preview
[[[DIVIDER]]]
Selection: context.txt names the checked preview
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
THE HANDOFF (97feb327; amended 2026-09-16): a complete, nonempty run has the
rider write adhocwalk.txt WHOLE beside the selected router, 0600,
os.replace'd -- the checked preview as the one uncommented line once DECANT
released it, the whole archive as a commented line beneath, and the archive
alone, labelled, when no preview was saved. IT REPLACES, IT NEVER MERGES, and
because `epr` now invites a hand-typed `!` line into this very file, it
REPLACES ONLY A FILE IT COULD HAVE WRITTEN: one uncommented absolute path to
an archive or a preview and nothing else. Any other uncommented line means a
human owns the file; the rider prints the new path lines and touches nothing,
and deleting the file lets the next walk rewrite it. Riding from a shell where
PIPULATE_ADHOC_FILE already names adhocwalk.txt refuses by name, non-fatally,
archive preserved, which is why cpr assigns in a subshell rather than
exporting. SELECTING IS NOT INCLUDING: naming adhocwalk.txt inside another
list includes its text and never recurses into its entries.
[[[DIVIDER]]]

THE HANDOFF (97feb327; amended 2026-09-16 and 2026-09-25): a complete,
nonempty run has the rider write context.txt WHOLE -- the one file
`context` opens and `compile` reads, PIPULATE_ADHOC_FILE or the repo root
-- 0600, os.replace'd: the checked preview as the one uncommented line once
DECANT released it, the whole archive as a commented line beneath, and the
archive alone, labelled, when no preview was saved. IT REPLACES, IT NEVER
MERGES, and because `context` invites a hand-typed line into this very
file, it REPLACES ONLY A FILE IT COULD HAVE WRITTEN: introduction.md plus
one uncommented absolute path to an archive or a preview and nothing else.
Any other uncommented line means a human owns the file; the rider prints
the new path lines and touches nothing, and deleting the file lets the next
walk rewrite it (witnessed 2026-09-25 against the operator's own working
list: two refusals, two paths printed, nothing written). SELECTING IS NOT
INCLUDING: naming context.txt inside another list includes its text and
never recurses into its entries.
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
The bridge to the issue text is a ! line in adhoc.txt.
[[[DIVIDER]]]
The bridge to the issue text is a ! line in context.txt.
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
environment under the skin. Concrete command-surface witness: `ahc` remains
the explicit ADHOC_CHOP author/workbench compiler, while `cpr` resolves the
walk router and invokes the same prompt_foo.py machinery through WALK_CHOP
with a lean frame, no tree, quiet output, and invocation-local
PIPULATE_ADHOC_FILE. The user names the goal; the framework stays behind it.
[[[DIVIDER]]]
environment under the skin. Concrete command-surface witness: `compile`
(ahc) is the workbench compiler, the walk writes the same context.txt that
`compile` reads, and the reading frame is one flag on the same command,
compile --frame lean --chop WALK_CHOP (2026-09-25). The user names the
goal; the framework stays behind it.
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
whole vocabulary is walk, plan, voice, epr, cpr and sources; warm and
[[[DIVIDER]]]
whole vocabulary is walk, plan, voice, context, prompt, compile and conn; warm and
[[[REPLACE]]]

Ignition: type exit, then ndq (the hook is copied into .git/hooks and g() is redefined on entry), and press F2 in any open Neovim or reopen it; the Python files load at call time, and Car 1's file is read at the next compile.

4. PROMPT

THE AFTER READING for the leftovers ride, deltas over absolutes, one probe at a time:

1. The adhoc census: prompt_foo.py 7 -> absent, flake.nix 6 -> absent, foo_files.py 18 -> 6 (the six survivors are dated: the 2026-09-21 receipt, the 2026-09-19 and 2026-09-16 TODOs, the DE-HARDWIRE TODO, the ahc-toggle TODO, the 2026-07-17 pin; a forget ride rules on them, not a rename), scripts/git_hooks/pre-commit 3 -> 1 (the for-loop line names all three), scripts/walk.py 1 -> absent, .gitignore 2 by design. Unchanged and next on the list: connectors (README plus seven files), GLOSSARY.md 3, the two SKILL.md files, scripts/two_arm.py, and imports/ascii_displays.py at 7, which is sealed art and needs a reseal, not a sed.
2. The export hunt: one file:line naming PIPULATE_ADHOC_FILE, or another empty set, in which case the next probe is `env -0 | tr '\0' '\n' | rg ADHOC` plus `rg -l PIPULATE_ADHOC_FILE /etc /run/current-system/etc 2>/dev/null`. The rename car (mv adhoc.txt to context.txt, rm adhocwalk.txt, edit the export, one gated paste, then exit and ndq) rides only once this line names the file.
3. The hook straddle: scripts copy 0 -> 3, .git/hooks copy 0 -> 3 after ignition; 3 and 0 means the shell was not re-entered.
4. The epr/cpr census: introduction.md 4 -> absent, prompt_foo.py 4 -> 1 (the 2026-09-18 conviction line), foo_files.py -3, scripts/mother_cat.py -1, scripts/boot_menu.py 1 both taps (the epitaph). Any hit under assets/installer or remotes/honeybot/www/npvg.org is a car for next turn, and the pad's is a nixops.sh truck, not a push.
5. The adhoc word count: scripts/walk.py 2 -> 1 (the adhoc_mutation JSON key stays until the ADHOC-name ruling), init.lua 2 -> 0, scripts/mother_cat.py 1 -> 0.
6. door 2: two lines -> door2=0.
7. The generated hook text: 0 -> 1 without ignition, because nix eval reads the dirty worktree; the function in your shell still needs exit then ndq.
8. The nvim gate: green both taps; a gate, not a landing.
9. The sound census: two .wav files under ~/.local/share/pipulate, the reader's path line in tools/scraper_tools.py, and .gitignore's *.wav line. Ruled: they move into assets/sounds/ with a README naming OpenGameArt 16323 and Freesound 611113 as CC0, .gitignore gets !assets/sounds/*.wav below its *.wav line, and the reader prefers the repo path with ~/.local/share as the fallback so the Mac stops being silent.
Car 1's receipt: HEADER_PRUNED with template=40 and tail_muted near 22 (13 files and 9 probe echoes), or HEADER_ALREADY_PRUNED if you had already cut it by hand.

NEXT TURN'S CARS, once the AFTER is in: the rename (probe 2 names the file), the sounds (probe 9, with tools/scraper_tools.py in context), and a ruling before any car on the walk writing into a human's file: the guard that refused today exists so a walk never overwrites your list, so the shape that keeps it is an append under a dated "# --- WALK <run_id> ---" marker with the vim lesson only when the file is absent, and a replace never; say yes or no and mother_cat.py is in context for it.
RULINGS CARRIED, not cars: whether the reading frame gets a word back or stays compile --frame lean; whether PIPULATE_ADHOC_FILE, ADHOC_CHOP, the ADHOC SLOT markers and walk.py's adhoc_mutation key are renamed, each a gated car with its readers in context; whether page 3 of the walk names prompt and compile beside its question; and the GLOSSARY.md purge, which is the operation's largest remaining bite and a ride of its own.

5. EXTERNAL DELIVERABLES

None this turn outside the repo.
MikeLev.in:
Same commands, run twice, one change between them. Where the readings differ is what the change did; the diff in the middle is the receipt.
1: Probe:
(nix) pipulate $ g

Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate ~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc /etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
for f in scripts/git_hooks/pre-commit .git/hooks/pre-commit; do echo "$f:$(rg -c 'context\.txt' "$f" || echo 0)"; done
rg -cw 'epr|cpr' --sort path prompt_foo.py foo_files.py flake.nix init.lua introduction.md GLOSSARY.md scripts assets/installer remotes/honeybot/www/npvg.org
for f in scripts/walk.py init.lua scripts/mother_cat.py; do echo "$f:$(rg -c adhoc "$f" || echo 0)"; done
rg -n 'door 2' scripts/sources_menu.py || echo door2=0
LD_LIBRARY_PATH="" nix eval --raw .#devShells.x86_64-linux.quiet.shellHook 2>/dev/null | rg -c 'Before conducting experiment' || echo 0
nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
ls -la ~/.local/share/pipulate/; rg -n 'local/share/pipulate|tick\.wav|ding\.wav|*\.wav' tools/scraper_tools.py .gitignore
./connectors/botify.py:1
./connectors/confluence.py:1
./connectors/gsc.py:1
./connectors/jira.py:1
./connectors/mcp.py:1
./connectors/README.md:1
./connectors/sheets.py:1
./connectors/slack.py:1
./connectors/wallet.py:3
./flake.nix:6
./foo_files.py:18
./.gitignore:2
./GLOSSARY.md:3
./imports/ascii_displays.py:7
./init.lua:2

./Notebooks/.agents/skills/gsc_readonly/SKILL.md:1
./Notebooks/.agents/skills/sheets_readonly/SKILL.md:1
./prompt_foo.py:7
./scripts/git_hooks/pre-commit:3
./scripts/two_arm.py:1
./scripts/walk.py:1
/home/mike/repos/pipulate/.env
12:PIPULATE_ADHOC_FILE="$HOME/.local/state/pipulate/adhoc.txt"
13:PIPULATE_ADHOC_FILE="$HOME/.local/state/pipulate/adhoc.txt"
ENV=/home/mike/.local/state/pipulate/adhoc.txt
scripts/git_hooks/pre-commit:0
.git/hooks/pre-commit:0
prompt_foo.py:5
foo_files.py:15
introduction.md:3
scripts/boot_menu.py:2
scripts/mother_cat.py:1
scripts/walk.py:2
init.lua:2
scripts/mother_cat.py:1
3:sources_menu.py — what door 2 opens onto.
5:boot_menu.py's door 2 drops the human into the Nix shell and tells them to
0
total 1076oaded successfully!
drwxr-xr-x 2 mike users   4096 Sep 19 07:17 .
drwx------ 42 mike users   4096 Sep 25 18:13 ..
-rw-r--r-- 1 mike users 444494 Sep 19 07:17 ding.wav
-rw-r--r-- 1 mike users 645230 Sep 19 07:17 tick.wav
.gitignore
13:*.wav

tools/scraper_tools.py
75:# Operator-owned courtesy sounds live outside the repo. tick.wav loops while
76:# the browser is opening; ding.wav plays once when the page-load phase ends.
79:SOUND_DIR = Path.home() / ".local/share/pipulate"
100:    args = _player_args(_sound_file("tick.wav"))
134:        args = _player_args(_sound_file("ding.wav"))
(nix) pipulate $

2: Context:
The Not-Managed-by-Git Safe-for-Client-Data place. Insert Simpson Couch Gag in white space below (explain anything to the audience you feel needs it explained)
_____   _   _
/ ____|___ _ __ | |_ _____ _| |_   Insert Simpson Couch Joke here.
| | / _ \| '_ \| __/ _ \ \/ / __| It's interesting shifting the identity here from adhoc.txt to context.txt.
| |__| (_) | | | | || __/> <| |_
________/|_| |_|_____/_/___|

```
#
# context.txt

# To Human: WELCOME TO VIM! It's really NeoVim but I say vim anyway.
# To AI: This is an alternate 40K view to the big book-ore rolling pin spine.

# 1. **Probe**: Baseline Reading
# 2. **Context**: Post-experiment *planned* reading instructions
# 3. **Patch**: The experiment and how to make it happen
# 4. **Prompt**: Post-experiment instructions and how to read results
# 5. **Deliverable**: How the world is forever different moving forward

# The first thing you need to know here is that everything that comes after the
# hash symbol (#) is commented out — and that's EVERYTHING in this file's default
# state. Begin editing-in lines for inclusion as part of the context or adding
# chunks of new context at the bottom. `Ctrl`+`v`, `j` (repeatedly), `l` (to move
# right), `d` (to delete). Reverse that with `Ctrl`+`v`, `j` (repeatedly),
# `Shift`+`i`, `# `, `Esc` to put the hashes back. You can just arrow-key around
# here with `h`, `j`, `k`, `l`. Save-and-quit is a bit tricky because another
# file is also loaded: `Esc`, `:`, `q`, `w`, `!`

# If this is stressing you out and you're a quitter and want to quit, just type:
# `Esc`, `:`, `q`, `!`, `Enter`. That will exit without saving any changes. If
# you want to get over this hump, type: `Esc`, `:`, `T`, `u`, `t`, `o`, `r`, `Enter`.

# This file is just to make it easy having options of what to edit into context.
# You can use whatever text-file you want to stack file-names and commands to
# build an output text-file with the identically stacked output of each file or
# command. In this way we vertically append or "stack" a bunch of text; simple as
# that. If you understand this concept, you're on your way to future-proofing
# yourself in the Age of AI. Congratulations! Here is how to include web pages:

#   !URL  ----------------------------------------------------------------------
#     when    Public page; what a stranger or crawler sees; the BEFORE of a
#             login-wall diagnosis
#     switch  It shows a login page -> `warm URL` once, then `?URL`
#
#   ?URL  ----------------------------------------------------------------------
#     when    Anything behind a login, on the site's persistent profile;
#             `check URL` first
#     switch  The lenses show a shell (nav, an `[Iframe]` leaf, no content) ->
#             read the wire truth for the XHR the frame makes, then call that
#             API with a connector
#
#   @URL  ----------------------------------------------------------------------
#     when    Every re-read of a page already scraped; no browser, no network
#     switch  The cached page is stale or was a login wall -> fresh `!` or `?`
#
```

```
#    $URL  ---------------------------------------------------------------
#      when    Exact markup: meta tags, a JSON blob in a `<script>`
#      note    Token-heavy; needs a prior scrape
#
#    %URL  ---------------------------------------------------------------
#      when    The network log distilled; SPA endpoint discovery
#      switch  It re-serves the wire truth you already have -> the API
#
#    ! cmd  --------------------------------------------------------------
#      when    Any bounded, non-interactive command as a live receipt
#      note    Cap it with `-n`; no aliases, no prompts
#
#    Connector  ----------------------------------------------------------
#      when    The number you want is one GET away
#      switch  LIST until the thing isn't in the list -> FETCH by id -> DRILL
#              the path the app's own frame called -> `--grep` to narrow a list
#              or find a leaf

# Every step is one argument longer than the last; the moment a lens shows less than the wire does is the moment to stop scraping.

# FOR 40K-FT VIEW (STORY & INFRASTRUCTURE) !!
# --- START EDITING-IN ON 1ST TURN ---

# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- ROLLING PIN that gives the 40K foot book-spine view of book-ore (only works for me because of local-only git repo)
# ~/repos/nixos/autognome.py  # <-- You wake up in the morning and your Tooling & Instrumentation folds out of you like Inspector Gadget.
# init.lua                  # <-- Those gadgets are made easy-to-use through nifty keyboard shortcuts (but ya gotta learn vim).
# assets/installer/install.sh # <-- Pipulate.com installer real home in github/pipulate repo
# # GLOSSARY.md               # <-- Like the back of a J.R.R. Tolkien book, there's kooky new terms to know.
# flake.nix                 # <-- Here is my hardware. Here is my state. Put on your sandbox. And please recreate. (Infrastructure as Code / IaC)
# prompt_foo.py             # <-- THIS system
# foo_files.py              # <-- main ROUTER
# requirements.in           # <-- We've "pinned" everything but still want a flexible Python Data Science virtualenv.
# pyproject.toml            # <-- How this is a citizen of the Python "pip install" ecosystem
# __init__.py               # <-- Version info

# --- END EDITING-IN ON 1ST TURN ---

# scripts/articles/lsa.py     # <-- 2ND BRAIN: Search external memory with `rgx`, `rgxc` & `posts` Blogging for Hackers Jekyll-compatible.

# OPTIONAL ACTUATORS (cheap and good to include to expand the AI's capabilities)
```

```
# cli.py                    # <-- Catch-all actuator for PyPI envs, Python anchoring, MCP tool-call (plus alternatives) and **kwargs like wrapping for CLI
# scripts/xp.py             # <-- Transforms host OS copy-paste buffer player-piano music into context-payload.
# scripts/ai.py             # <-- How I constantly use local AI to write git commit messages with `m` alias.
# scripts/crawl.py          # <-- Feel free to ask for something to be crawled and included in the next turn.
# scripts/weblogin.py       # <-- Lets the user "warm up" the cache for their web logins at their leisure on a profile that persists.
# scripts/webclip_2_markdown.py  # <-- Surprisingly important program.

# MISCELLANEOUS (rare to include but sometimes critical)
# scripts/foo_cartridge.py    # Needs description
# scripts/foo_replay.py       # Needs description
# release.py                  # <-- How everything ends up where it does (GitHub, PyPI, etc.)
# imports/voice_synthesis.py  # <-- The wand can talk to you
# imports/ascii_displays.py   # <-- Where all the ASCII Art lives
# scripts/release/version_sync.py  # <-- Needs to be wrapped into release.py and eliminated, I think.

#                   --- Under this line is were you paste what the AI gives you ---
#                   --- We call it context but it's really just the right-hand  ---
#                   --- blast-radius of the "probes" to make this all science.  ---

# Carry-over as the important work-in-progress parts of the project here just
# like above but not as long-standing overarching to the framework but rather
# for the current hot spots actively being worked on.

# --- START THIS DISCUSSION ---

# Get things started here! Guess at what context should be included.
# If you get it wrong, you're just wasting 1-turn because the AI will help.
# Un-comment lines, add lines with absolute-path filenames or `! ` commands.

# Context 1 (Edit-in selections from above and add new files immediately below)
# # LAUNCH
# walk                      # <-- Repo-root wrapper; delegates to mck.sh, NOT to scripts/walk.py
# assets/installer/mck.sh     # <-- THE LAUNCHER (curl -fsSL pipulate.com/mck.sh | bash): finds the workshop, forces the spoken rehearsal on first contact, asks for the word RIDE
# scripts/mother_cat.py       # <-- THE RIDER (alias: mothercat), Car B: actuates walk.py's validated plan; per-run capture banking, DECANT, the adhocwalk.txt writer
# scripts/boot_menu.py        # <-- The one-door command list; short on shell entry, expanded by `all`, both rendered from tuples in the file
# scripts/sources_menu.py     # <-- The roster `conn` prints; filename stays descriptive, typed word is shorter
#
# # COMPILE
```

scripts/walk_compile.py   # <-- .walk.md -> <name>.yaml BESIDE the surface, and only where git ignores it; url_env stops only (it refuses a scheme separator anywhere in its output), so it is the PRIVATE lane's compiler and a bundled trail in assets/trails/ is hand-written JSON; refuses on every TODO left in the surface; stdlib only, imports nothing
#
VALIDATE
scripts/walk.py   # <-- Car A, the planner: loads a trail, refuses unknown or missing keys in BOTH directions, prints the dry-run plan; holds DEFAULT_TRAIL; never actuates
#
SEAL
scripts/walk_cartridge.py   # <-- trail -> data/walks/<sha256>/walk.zip (trail + manifest with hash and consent surface); reseal is byte-identical
#
CAPTURE
scripts/weblogin.py   # <-- SETTLE ahead of time: warm a login on the persistent profile; --profile default unless told otherwise
tools/scraper_tools.py   # <-- The browser capture and the CAPTURE fence (_capture_checkpoint is a write barrier, not a prompt)
imports/voice_synthesis.py # <-- Piper: scripted narration, never a model (ATTRIBUTED-VOICE)
#
HAND OFF
prompt_foo.py   # <-- Compiles the selected router (adhocwalk.txt: `epr` edits it, `cpr` compiles it) into a payload
scripts/foo_cartridge.py   # <-- The second seal, over the EVIDENCE: payload.md, prompt.md, manifest.json
scripts/foo_replay.py   # <-- Context extraction and attention checks; not browser or API replay
tests/test_mck_rep2.py   # <-- Rep 2: the earmark's owed side-by-side witness
#
THE TRAILS (bundled; each status is the newest receipt, never a promise)
assets/trails/public_walk.json   # profile default; SETTLE trivial; three unlinked npvg.org pages (the_word, the_receipt, the_two_pages), inline script on stop three only, connector noop.py; RIDDEN 2026-09-15 on these stops, 3 of 3 at artifacts=11 each, archive complete, DECANT AUTHORIZED at 20,992 B with checks 0/0/0, and the chatbot given stop three's question said the preview does not carry the server's sentence, which is true (the diff-lens TODO); the 2026-08-01 ride walked the OLD stop set (example.com, mikelev.in, pipulate.com); what bare `walk` rides
assets/trails/practice.yaml   # one configurable page; UNREAD, label stale since 2026-08-01
assets/trails/first_context.yaml   # profile default; SETTLE real; Jira/Botify/Gmail, THREE wallet kinds in one trail; UNRIDDEN
assets/trails/botify_pageworkers.yaml # profile botify; SETTLE real; UNRIDDEN -- the only trail on a non-default profile, and no such ride has ever been witnessed
assets/trails/jira_for_you.yaml   # profile default; SETTLE real (Atlassian sign-in); top of the ticket loop; RIDDEN 2026-09-01 and DECANTED, the sign-in moment unrecorded (a cold profile records it)
assets/trails/ticket.yaml   # profile default; SETTLE real; url_env x2 (JIRA, BOTIFY) fed

by one gitignored exports file per ticket under Notebooks/Client_Work/tickets/; stop two of the ticket loop; RIDDEN 2026-09-01 to stop 1 then REFUSED at stop 2 on an unset url_env, Linux and Mac -- the ride that bought the rider its PRE-FLIGHT. RULING: the connector does NOT run at DECANT (DECANT gates only CAPTURE-fenced material; API auth is a different wallet; the rider never harvests). The bridge to the issue text is a ! line in adhoc.txt.
assets/trails/se_ticket.yaml # profile default; SETTLE real; the SE ticket template: for_you (literal) + issue (JIRA, required) + project/slack/confluence (optional, skipped when unset); RIDDEN 2026-09-02, 3 of 5 captured, 2 skipped, 72,383 B decanted, card showed both rows; harvest regexes on optional stops are PLACEHOLDERS (botify_analysis's already fullmatched a live analysisSlug); guidance strings hand-count "of five" and go stale on the sixth stop
#
UNLINKED PAGES (npvg.org; nothing links to them and no page links out, so a trail is the only edge; nixops.sh rsyncs them, no rebuild)
remotes/honeybot/www/npvg.org/index.html # <-- the door: a browser gets this page, curl and wget get install.sh ($npvg_index)
remotes/honeybot/www/npvg.org/walk/1/index.html # <-- public_walk stop one: the capture word; no script, so source and hydrated DOM should match
remotes/honeybot/www/npvg.org/walk/2/index.html # <-- stop two: count the archive's fingerprints against the terminal's artifacts= number
remotes/honeybot/www/npvg.org/walk/3/index.html # <-- stop three: the walk's only script rewrites the server's sentence and appends a paragraph; the DECANT test lives here
#
OFF-ROSTER DISTRIBUTION RESIDUE (not a walk dependency)
assets/installer/replay.sh # <-- OFF the roster 2026-08-01, stranded; re-add needs syntax + one ride + a pinned verifier fetch

Context 2
--- THIS RIDE: the files the cars touched, so the AFTER can read them ---
.gitignore
prompt_foo.py
apply.py
scripts/mother_cat.py
scripts/boot_menu.py
init.lua
flake.nix
foo_files.py
--- NEXT-TURN CARS: the rename's remaining homes; the census names which ---
scripts/xp.py
scripts/git_hooks/pre-commit
introduction.md
scripts/walk.py
scripts/sources_menu.py
GLOSSARY.md
README.md
AGENTS.md
--- THE 40K-FOOT VIEW (your call) ---
! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs

```
# # --- PROBE ECHOES ---
# ! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
# ! rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
# ! .venv/bin/python scripts/boot_menu.py --recall
# ! rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type
context' scripts/mother_cat.py
# ! rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
# ! awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END
.(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
# ! git check-ignore -v context.txt || echo NOT_IGNORED
# ! for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f
"$f" && rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
# ! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2

# Context 3
# --- NEXT TURN'S CARS: the rename (once line 2 names the export), the sounds, the walk's
write ---
scripts/mother_cat.py
tools/scraper_tools.py
.gitignore
# flake.nix        # <-- only if Car 8 or 9 refused; line 7 is their witness
# foo_files.py     # <-- only if Car 10 refused, or to rule on the six dated adhoc lines it keeps
# init.lua         # <-- only if Car 6 refused
# introduction.md  # <-- only if Car 3 refused
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
! for f in scripts/git_hooks/pre-commit .git/hooks/pre-commit; do echo "$f:$(rg -c 'context\.txt' "$f"
|| echo 0)"; done
! rg -cw 'epr|cpr' --sort path prompt_foo.py foo_files.py flake.nix init.lua introduction.md
GLOSSARY.md scripts assets/installer remotes/honeybot/www/npvg.org
! for f in scripts/walk.py init.lua scripts/mother_cat.py; do echo "$f:$(rg -c adhoc "$f" || echo 0)";
done
! rg -n 'door 2' scripts/sources_menu.py || echo door2=0
! LD_LIBRARY_PATH="" nix eval --raw .#devShells.x86_64-linux.quiet.shellHook 2>/dev/null |
rg -c 'Before conducting experiment' || echo 0
! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
! ls -la ~/.local/share/pipulate/; rg -n 'local/share/pipulate|tick\.wav|ding\.wav|\*\.wav'
tools/scraper_tools.py .gitignore
```

3: Patches:
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

```
nothing to commit, working tree clean
(nix) pipulate $ python - <<'EOF'
import os, sys
from pathlib import Path
root = Path(os.environ.get("PIPULATE_ROOT", "."))
sel = os.environ.get("PIPULATE_ADHOC_FILE", "")
p = Path(sel).expanduser() if sel.strip() else root / "context.txt"
if not p.is_file():
    sys.exit(f"STOP: no context file at {p}")
lines = p.read_text(encoding="utf-8").split("\n")
if not lines[0].startswith("# AD HOC CHOP"):
    print(f"HEADER_ALREADY_PRUNED {p} first_line={lines[0][:50]!r}; nothing written")
    sys.exit(0)
C = [i for i, l in enumerate(lines) if l.startswith("# Context 1 (Edit-in")]
assert len(C) == 1, ("STOP: Context 1 marker", C)
ff = (root / "foo_files.py").read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(ff) if l.rstrip() == "# --- START `context.txt` TEMPLATE ---"]
E = [i for i, l in enumerate(ff) if l.rstrip() == "# --- END `context.txt` TEMPLATE ---"]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: template markers", S, E)
template = ff[S[0] + 1:E[0]]
while template and not template[-1].strip():
    template.pop()
tail, muted = [], 0
for l in lines[C[0] + 1:]:
    if l.strip() and not l.lstrip().startswith("#"):
        l, muted = "# " + l, muted + 1
    tail.append(l)
out = template + ["", "# Context 1"] + tail
p.write_text("\n".join(out), encoding="utf-8")
print(f"HEADER_PRUNED {p} header_lines_replaced={C[0] + 1} template={len(template)}
tail_kept={len(tail)} tail_muted={muted}")
EOF
HEADER_ALREADY_PRUNED /home/mike/.local/state/pipulate/adhoc.txt first_line='# The
Not-Managed-by-Git Safe-for-Client-Data plac'; nothing written
(nix) pipulate $ d
(nix) pipulate $ context
(nix) pipulate $ # Oh, I anticipated that I think.
(nix) pipulate $
```

Whoops, my bad. Okay, onto the traditional patches.

```
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/git_hooks/pre-commit'.
```

```
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/git_hooks/pre-commit'.
(nix) pipulate $ d
diff --git a/scripts/git_hooks/pre-commit b/scripts/git_hooks/pre-commit
index 17ddf58f..7f129aae 100644
--- a/scripts/git_hooks/pre-commit
+++ b/scripts/git_hooks/pre-commit
@@ -32,12 +32,14 @@
 # worktree: (1) the gitignored overlay must never be force-added;
 # (2) the tracked ADHOC socket in foo_files.py must stay structurally
 # empty (blank lines and comments only).
-if git ls-files --error-unmatch -- adhoc.txt >/dev/null 2>&1; then
-    echo "🛑 COMMIT BLOCKED: adhoc.txt is tracked in the index."
-    echo "   The adhoc overlay must never enter history."
-    echo "   Fix: git restore --staged adhoc.txt   (or git rm --cached adhoc.txt)"
-    exit 1
-fi
+for overlay in context.txt adhoc.txt adhocwalk.txt; do
+    if git ls-files --error-unmatch -- "$overlay" >/dev/null 2>&1; then
+        echo "🛑 COMMIT BLOCKED: $overlay is tracked in the index."
+        echo "   The context file (and its two retired names) must never enter history."
+        echo "   Fix: git restore --staged $overlay   (or git rm --cached $overlay)"
+        exit 1
+    fi
+done
 if git show :foo_files.py 2>/dev/null | awk '
     /^# --- ADHOC SLOT START ---$/ {inside=1; next}
     /^# --- ADHOC SLOT END ---$/   {inside=0; next}
@@ -46,7 +48,7 @@ if git show :foo_files.py 2>/dev/null | awk '
     echo "🛑 COMMIT BLOCKED: staged foo_files.py carries live payload in the ADHOC slot."
     echo "   The tracked socket must contain only comments; contents belong in the"
-    echo "   gitignored overlay (edit with ahe)."
+    echo "   gitignored context file (edit with context)."
     exit 1
 fi

(nix) pipulate $ m
📝 Committing: fix: block commits with tracked context files
[main e637f170] fix: block commits with tracked context files
 1 file changed, 9 insertions(+), 7 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'introduction.md'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'introduction.md'.
(nix) pipulate $ d
diff --git a/introduction.md b/introduction.md
index ac7021f7..1a67f70d 100644
--- a/introduction.md
```

```
+++ b/introduction.md
@@ -26,20 +26,22 @@ A checked, trimmed summary of them, decant-preview.md, follows this file.

 ## What the person can do next

-They edit a plain text file called the prompt router by typing epr. Every
+They edit a plain text file called context.txt by typing context. Every
  line in it that does not start with # is one thing you will see the next
  time they compile: a file path, or a shell command written after `! ` whose
-output rides along. They compile it by typing cpr and paste the result here
-again. The full archive, captures.md, is named in that file but commented
-out, because it is unreviewed and may hold private details.
+output rides along. They save a question from their clipboard by typing
+prompt, build the file and the question into one payload by typing compile,
+and paste the result here again. The full archive, captures.md, is named in
+that file but commented out, because it is unreviewed and may hold private
+details.

 ## How to help

  Answer from the sections that follow this file, and say which section
  supports each claim. If the summary does not carry what a question needs,
  say so plainly and name where it would be found: a lens in the full archive,
-a fresh walk, or a router line. When you suggest adding something to the
-router, give the exact line to type, and say whether the person should read
-that file themselves before compiling it. Use only the words they have
-already met at the terminal (menu, walk, epr, cpr) and in the editor (j, k,
-Esc, :q, :q!, :wq). Do not ask them to install anything.
+a fresh walk, or a line in context.txt. When you suggest adding something to
+context.txt, give the exact line to type, and say whether the person should
+read that file themselves before compiling it. Use only the words they have
+already met at the terminal (menu, walk, context, prompt, compile) and in the
+editor (j, k, Esc, :q, :q!, :wq). Do not ask them to install anything.
(nix) pipulate $ m
📝 Committing: chore: Update introduction.md with refined instructions and terminology
[main 24ba3e80] chore: Update introduction.md with refined instructions and terminology
 1 file changed, 11 insertions(+), 9 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/walk.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/sources_menu.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/sources_menu.py'.
(nix) pipulate $ d
diff --git a/scripts/sources_menu.py b/scripts/sources_menu.py
index ea6811d6..fbf7ff95 100644
--- a/scripts/sources_menu.py
+++ b/scripts/sources_menu.py
```

```
@@ -1,9 +1,9 @@
 #!/usr/bin/env python3
 """
-sources_menu.py — what door 2 opens onto.
+sources_menu.py — what `conn` prints.

-boot_menu.py's door 2 drops the human into the Nix shell and tells them to
-type `sources`. This is the roster they get: the commands that reach OUTSIDE
+boot_menu.py lists `conn` at every shell entry and the Nix shell defines it.
+This is the roster it prints: the commands that reach OUTSIDE
 this machine, each row's description GENERATED from the target script's own
 module docstring rather than hand-authored here.

diff --git a/scripts/walk.py b/scripts/walk.py
index 0386cf11..737dd68c 100644
--- a/scripts/walk.py
+++ b/scripts/walk.py
@@ -3,7 +3,7 @@

 Trail files use JSON. That keeps this car stdlib-only, duplicate-key-checkable,
 and explicit about the syntax humans are editing. There is deliberately no
-browser, voice, shell, or adhoc.txt mutation path in this file.
+browser, voice, shell, or context.txt mutation path in this file.
 """


 import argparse
(nix) pipulate $ m
📝 Committing: chore: Update documentation for sources_menu.py and walk.py
[main 80be729c] chore: Update documentation for sources_menu.py and walk.py
 2 files changed, 4 insertions(+), 4 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
(nix) pipulate $ d
diff --git a/scripts/mother_cat.py b/scripts/mother_cat.py
index 39f43a75..d2463c82 100644
--- a/scripts/mother_cat.py
+++ b/scripts/mother_cat.py
@@ -332,7 +332,7 @@ def _bank_capture(archive, trail, index, stop, params, result):
     return problems


-# THE TRAINING WHEELS (2026-09-20, read off the first Mac epr screen). `epr`
+# THE TRAINING WHEELS (2026-09-20, read off the first Mac epr screen). `context`
```

```
 # is a newcomer's first vim, and the router is the one place the lesson can
 # sit where their eyes already are, so the first router a machine ever
 # writes carries it. The keys are the bridge from less: j and k are the same
@@ -427,7 +427,7 @@ def _write_walk_router(archive_path, preview_path=None, first=None):
         if owned and (len(routed) != 1 or not routed[0].startswith("/")
                       or not routed[0].endswith(("captures.md", "decant-preview.md"))):
             raise ValueError(
-                "hand-edited router kept, nothing written; add the line(s) yourself: "
+                f"{target.name} has lines of your own, so nothing was written; add the line(s)
yourself: "
                 + " ".join(line for line in (preview, source) if line)
                 + " (or delete the file and the next walk rewrites it)")
     if first is None:
@@ -459,7 +459,7 @@ def _write_walk_router(archive_path, preview_path=None, first=None):
     try:
         with tempfile.NamedTemporaryFile(
             mode="w", encoding="utf-8", newline="\n", dir=target.parent,
-            prefix=".adhocwalk-", delete=False,
+            prefix=".context-", delete=False,
         ) as stream:
             temp = Path(stream.name)
             os.fchmod(stream.fileno(), 0o600)
@@ -492,7 +492,7 @@ def _finish_capture_archive(archive, status, skipped=()):
             target = _write_walk_router(archive["path"])
         except Exception as exc:
             print(f"  CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
-            print("  Archive preserved; the router on disk is unchanged.")
+            print("  Archive preserved; the context file on disk is unchanged.")
         else:
             print(f"  CONTEXT FILE  {target}  (0600; lists the UNSANITIZED archive until a
summary is saved)")
     else:
@@ -1025,7 +1025,7 @@ async def _ride_steps(trail_path, archive, dry_narrate=False,
exports_path=None,
         def checkpoint_narration():
             nonlocal disclosed
             disclosed = _narrate(
-                "The page has settled. Come back to the terminal and type CAPTURE.",
+                "The page has loaded. Come back to the terminal and type CAPTURE.",
                 disclosed,
             )

(nix) pipulate $ m
📝 Committing: chore: Update context file prefix to ".context-"
[main b821f6c8] chore: Update context file prefix to ".context-"
 1 file changed, 5 insertions(+), 5 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
```

```
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'init.lua'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'init.lua'.
(nix) pipulate $ d
diff --git a/init.lua b/init.lua
index 6b1fa9b7..4967fea6 100644
--- a/init.lua
+++ b/init.lua
@@ -329,7 +329,7 @@ end

 function toggle_text_width()
    -- THE MARGIN RELEASE: flip between tw=80 (journal prose auto-wraps at
-    -- the column) and tw=0 (long lines run free for adhoc.txt masthead art
+    -- the column) and tw=0 (long lines run free for context.txt lines
    -- and other wide-format work). One key, no :set incantation, and the
    -- notify always tells you which regime you are typing in.
    if vim.opt.textwidth:get() == 0 then
@@ -785,7 +785,7 @@ function hop_off_sandworm()
        .. "   SEARCH anchor is not banked; it is a hand edit the operator will\n"
        .. "   not make. These BANK cars are the ONLY patches a dismount emits.\n"
        .. "3. DANGLING: what carries forward unbanked? One line each, no essays.\n"
-        .. "4. SEED: the adhoc.txt lines (and TODO_SLUGS if narrative context is\n"
+        .. "4. SEED: the context.txt lines (and TODO_SLUGS if narrative context is\n"
        .. "   needed) for the next ride's first compile.\n"
        .. "5. CLOSING: a closing summary for the BOTTOM of the article — the\n"
        .. "   final take-away, tied to the book's larger arc where it fits\n"
(nix) pipulate $ m
📝 Committing: chore: Update text width toggle comment in init.lua
[main ff9048d0] chore: Update text width toggle comment in init.lua
 1 file changed, 2 insertions(+), 2 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'prompt_foo.py'.
(nix) pipulate $ d
diff --git a/prompt_foo.py b/prompt_foo.py
index c92a2fdc..b228ef68 100644
--- a/prompt_foo.py
+++ b/prompt_foo.py
@@ -1555,8 +1555,8 @@ class PromptBuilder:
        self.tool_roster_content = tool_roster_content
```

```
        # THE FRAME IS NAMED BY THE CALLER (2026-09-19). "full" is every
        # existing compile: the full checklist and the five-car train ride
-       # ahead of the prompt. "lean" is the reading frame cpr passes, for a
-       # question asked of a walk preview. Nothing reads the prompt text to
+       # ahead of the prompt. "lean" is the reading frame for a question asked
+       # of a walk preview (compile --frame lean). Nothing reads the prompt text to
        # guess which; a chop selects files and a frame selects the checklist,
        # and the two are separate flags on purpose.
        self.frame = frame
@@ -1793,7 +1793,7 @@ class PromptBuilder:
        cars because the checklist rode with it. The routing invariant and
        the live-receipts clause survive here because they describe the
        artifact, and the artifact is the same either way; the patch protocol
-       and the train do not, because nothing in a cpr compile is asking for
+       and the train do not, because nothing in a lean compile is asking for
        a repository change.
        """
        return '''# ⚠️ ROUTING INVARIANT: Read this section before acting on anything.
@@ -2459,7 +2459,7 @@ def update_agents_md_in_place():

    ORDERING NOTE (why the straddle closes in ONE compile): main() calls this
    at step 2, BEFORE the `!` executor loop runs. A probe echoed into
-   adhoc.txt therefore witnesses THIS compile's fill, not the previous one --
+   context.txt therefore witnesses THIS compile's fill, not the previous one --
    the opposite of the foo.zip DOUBLE-TAP lag, and the reason the sentinels
    were landed EMPTY. A hand-copied frame could never prove the generator ran.
    """
@@ -2531,7 +2531,7 @@ def update_readme_md_in_place():
    project has. A stale README is a wound; a confidently wrong one is a lie.

    ORDERING: called from main() at step 2, alongside the AGENTS.md splice and
-   BEFORE the `!` executor loop -- so a probe echoed into adhoc.txt witnesses
+   BEFORE the `!` executor loop -- so a probe echoed into context.txt witnesses
    THIS compile's fill rather than the previous one's.
    """
    readme_path = os.path.join(REPO_ROOT, "README.md")
@@ -2650,7 +2650,7 @@ def check_topological_integrity(chop_var: str =
"AI_PHOOEY_CHOP", format_kwargs:
            # '<--' note, or a two-space '#' inline note follows it. A comment
            # that continues with one space + words is a sentence, not a path:
            # a MIME type (text/markdown), a protocol name (SEARCH/REPLACE), a
-           # filename cited mid-thought (index.md, SKILL.md, adhoc.txt), or a
+           # filename cited mid-thought (index.md, SKILL.md, context.txt), or a
            # paren-glued token ((payload.md). Those minted the phantom
            # Broken-References alert; the compiler's own parser never loads a
            # '#' line, and now neither does this checker's prose.
@@ -2708,8 +2708,8 @@ def main():
    """Main function to parse args, process files, and generate output."""
```

```
    global VERBOSE
    # THE OUROBOROS LOCK (convicted 2026-07-22, Ctrl+C receipt in-compile):
-   # a `! ... prompt_foo.py ...` line in adhoc.txt makes the compiler run a
-   # probe that runs the compiler that splices the same adhoc.txt — quine
+    # a `! ... prompt_foo.py ...` line in context.txt makes the compiler run a
+    # probe that runs the compiler that splices the same context.txt — quine
    # recursion until timeout cascade or human interrupt. Env vars inherit
    # through the `!` executor's Popen, so a nested invocation sees the lock,
    # emits a one-line receipt to stdout, and exits 0. The receipt lands in
@@ -2861,9 +2861,9 @@ def main():
    parser.add_argument('-v', '--verbose', action='store_true', help='Echo progress
announcements (step headers, flags echoed back) that the Rule of Silence hides by default.
Readings, receipts, and gates always print.')
    parser.add_argument('--chop', type=str, default='AI_PHOOEY_CHOP', help='Specify an
alternative payload variable from foo_files.py')
    # THE FRAME IS A FLAG, NOT A PROPERTY OF THE CHOP (2026-09-19). A chop
-   # selects files; a frame selects what rides ahead of the prompt. cpr in
-   # flake.nix passes --frame lean; every other alias omits it and gets the
-   # default, so no existing compile changes shape. The TODO that seeded this
+    # selects files; a frame selects what rides ahead of the prompt. Since
+    # 2026-09-25 no alias passes it: compile --frame lean is the spelling, and
+    # every compile keeps the default until a hand types it. The TODO that seeded this
    # refused choosing by prompt text: the caller names the lane.
    parser.add_argument('--frame', type=str, choices=('full', 'lean'), default='full', help='What rides
ahead of the prompt: full is the complete checklist and five-car train (the default); lean is a
reading frame for a question asked of a walk preview; pass it by hand as compile --frame lean.')
    parser.add_argument('--bumper', type=str, default=None, help='Inject a pre-registered
bumper matrix from flippers.json (e.g., gold, cat)')
@@ -3097,7 +3097,7 @@ def main():
                    content += f"\n--- stderr ---\n{_err}"

                processed_files_data.append({
-                    # Marker parity: the payload label IS the adhoc.txt line.
+                    # Marker parity: the payload label IS the context.txt line.
                    # `--- START: ! rgxc foo ---` greps identically to the
                    # `! rgxc foo` that summoned it, for humans and models alike.
                    "path": f"! {raw_command}", "comment": comment, "content": content,
@@ -3334,7 +3334,7 @@ def main():
        path = os.path.expanduser(path)
        full_path = os.path.join(REPO_ROOT, path) if not os.path.isabs(path) else path
        # THE DE-PREFIXED COMMAND HINT (convicted 2026-08-03): three probe
-       # echoes lost their leading "! " during a hand-copy into adhoc.txt, so
+        # echoes lost their leading "! " during a hand-copy into context.txt, so
        # the compiler parsed them as PATHS, printed three IDENTICAL generic
        # warnings, and the operator's eye slid past all three -- costing the
        # AFTER half of an exit-code straddle. THE DISCRIMINATION QUESTION,
(nix) pipulate $ m
📝 Committing: refactor: update prompt builder frame selection to lean compile context.txt
```

```
[main 51d948d3] refactor: update prompt builder frame selection to lean compile context.txt
 1 file changed, 13 insertions(+), 13 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
(nix) pipulate $ d
diff --git a/flake.nix b/flake.nix
index 659c18f4..811a80f1 100644
--- a/flake.nix
+++ b/flake.nix
@@ -194,7 +194,7 @@
         );

         # Real commands, not shell functions, so Python-spawned `!` commands in
-        # adhoc.txt inherit them through PATH just like interactive Bash does.
+        # context.txt inherit them through PATH just like interactive Bash does.
         postsCommand = pkgs.writeShellScriptBin "posts" ''
           set -euo pipefail
           root=""${PIPULATE_ROOT:-$PWD}"
@@ -223,7 +223,7 @@
         # work in one context window.
         #
         # A COMMAND, NOT A FUNCTION, for the same reason `posts` is one: a `!`
-        # chisel-strike in adhoc.txt spawns a non-interactive child that
+        # chisel-strike in context.txt spawns a non-interactive child that
         # inherits PATH and never inherits functions, so `! postsc 5` resolves
         # and `! posts2 5` cannot. Forwarding is total, so `-t`, `--reverse`,
         # `--match`, `--slugs` and the bare-N positional all still work, and
@@ -1419,7 +1419,7 @@ runScript = pkgs.writeShellScriptBin "run-script" ''
           # THE CONNECTOR GRAMMAR (idea #7 made literal): tiny Unix commands,
           # one per API, each a self-contained file in connectors/.
           # Args pass through: `botify org/project`, `confluence ENG`,
-          # `gmail <thread_id>`. Interactive-shell only — adhoc.txt `!` lines
+          # `gmail <thread_id>`. Interactive-shell only — context.txt `!` lines
           # ride as DERIVATIONS in commonPackages above, so the short spelling
           # is legal everywhere -- a router line is now `! jira SWCX-1234`.
           # THE SEVEN LEFT THIS BLOCK 2026-09-16 for connectorCommands above.
@@ -1563,7 +1563,7 @@ runScript = pkgs.writeShellScriptBin "run-script" ''
           # documented miss, not a bug: prompt_foo prints the ledger-miss line
           # and names the ! scrape. So: sniff -f once, sniff freely after.
           #
-          # Lines ACCUMULATE on purpose. adhoc.txt is a scratchpad, `ahe` is
+          # Lines ACCUMULATE on purpose. context.txt is a scratchpad, `context` is
```

```
          # the pruner, and every sigil line re-fires on every later compile.
          # This function only appends, and prints exactly what it wrote.
          sniff() {
@@ -1605,7 +1605,7 @@ runScript = pkgs.writeShellScriptBin "run-script" "
          # /bin/sh via Popen(shell=True). Any `!` probe of this spec returns
          # the SAME answer whether the spec exists or not -- a false receipt in
          # both directions -- so it is deliberately never echoed into
-         # adhoc.txt. The only honest witness is a human pressing Tab.
+          # context.txt. The only honest witness is a human pressing Tab.
          #
          # Bare domains only, no scheme: COMP_WORDBREAKS contains ':', so
          # typing "https://exa<TAB>" splits the word and completion cannot see
@@ -1830,7 +1830,7 @@ print(max(1, n))
          # slugs-ordered preserves input order for narrative control
          sluggo() { for slug in "$@"; do (cd "$PIPULATE_ROOT" && python scripts/articles/lsa.py
-t 1 --match "$slug" --fmt paths); done; }
          # `rgx` and `rgxc` are Nix-packaged commands above, not shell
-         # functions, so interactive use and adhoc.txt child shells share one implementation.
+          # functions, so interactive use and context.txt child shells share one implementation.
          # UNNAMED-ROOT RULE conviction: this invoked release.py through the
          # CWD while `smart` invokes the SAME SCRIPT wrapped in
          # cd "$PIPULATE_ROOT" -- one script, two spellings, one anchored.
(nix) pipulate $ m
📝 Committing: chore: refine documentation for `context.txt` and `adhoc.txt` inheritances
[main 0b59b45e] chore: refine documentation for `context.txt` and `adhoc.txt` inheritances
 1 file changed, 6 insertions(+), 6 deletions(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'flake.nix'.
(nix) pipulate $ d
diff --git a/flake.nix b/flake.nix
index 811a80f1..580127e5 100644
--- a/flake.nix
+++ b/flake.nix
@@ -1853,8 +1853,17 @@ print(max(1, n))
          # different first line from each after the repaint, so a copy taken
          # from the top of the screen named which one had run. One body,
          # one first line, typed the same as before.
+          # THE FIRST LINE IS A VERDICT (2026-09-25, operator's ruling): the
+          # boundary is a commit, so a dirty tree has none yet and the line
+          # says what makes one; a clean tree has it and the line says so.
          g() {
-           clear -x && echo "Left-hand causal boundary \"blast radius\" established. Make 1
change and test." && git status
+            clear -x
+            if [ -n "$(git status --porcelain)" ]; then
+              echo "Before conducting experiment, type \"blast\" to establish left-hand causal
boundary."
```

```
+          else
+            echo "Left-hand causal boundary \"blast radius\" established. Make 1 change and test."
+          fi
+          git status
        }
        m() {
          # THE UNTRACKED-FILE DEBT (banked TODO 2026-07-20, discharged
(nix) pipulate $ m
📝 Committing: chore: Add git status check for blast radius boundary
[main faa489aa] chore: Add git status check for blast radius boundary
 1 file changed, 10 insertions(+), 1 deletion(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
(nix) pipulate $ d
diff --git a/foo_files.py b/foo_files.py
index a2704860..cb4ace9d 100644
--- a/foo_files.py
+++ b/foo_files.py
@@ -205,7 +205,7 @@ AI_PHOOEY_CHOP = r"""#
 # § THE KEY/VALUE CONTRACT (banked 2026-08-30) -- a router line opening with "# §" is a KEY: handle, date, one imperative, one physical line; its VALUE (mechanism, conviction, siblings, debts) lives in GLOSSARY.md under the same handle (rg -in '<handle>' GLOSSARY.md). Four fates: a RULE graduates to a key in place; a RECEIPT fades off the capped newest-first block; an EARMARK is a value without a key and either graduates or demotes to a todo; a TODO stays one line and dies when done. THE KEY TEST: would this line, read cold, have prevented the conviction that banked it? Family rosters ride the parent key, never every child.
 # § THE PROBE ECONOMY RULE -- a probe is cheap only when its output is bounded: cap or measure with wc -l, head, tail, rg -l, or an explicit limit before it rides the ledger; unbounded stdout is a context import, not a probe.

-# § THE PROBE ECHO RULE -- every probe recommended for hand-execution is also echoed verbatim as a `!` line into the next adhoc.txt; the hand-run is BEFORE, the compiled re-run is AFTER: one probe, two receipts, straddling the patch.
+# § THE PROBE ECHO RULE -- every probe recommended for hand-execution is also echoed
```

verbatim as a `!` line into the next context.txt; the hand-run is BEFORE, the compiled re-run is AFTER: one probe, two receipts, straddling the patch.
 # § NEXT CONTEXT IS THE WHOLE LIST (banked 2026-09-20, amended 2026-09-21) -- name in full, in NEXT CONTEXT, every file a next-turn car will patch: the ones that rode this compile, the ones that ride every compile, and the ones whose car did NOT land, because the payload a model reads is the only source it may patch and "already there" is a claim about the past. Value: the header comment in apply.py; GLOSSARY.md entry owed.
 # § THE KATA'S NAME (banked 2026-07-17) -- Probe, Patch, Prompt: hand-run receipts before, human-actuated mutation during, pre-loaded compile after; titles and section headers say it too. Value: the vocabulary entry Probe / Patch / Prompt in GLOSSARY.md.
 # ONE-LINER COROLLARY (banked 2026-07-19): a `!` line is ONE shell command.
@@ -284,7 +284,7 @@ AI_PHOOEY_CHOP = r"""#
 # receipts, never memory; (2) BANK graduations as SEARCH/REPLACE patch
 # cars, deletions included; (3) name the DANGLING carried forward
 # unbanked, one line each; (4) SEED the next ride's first compile with
-# adhoc.txt lines. Mechanism: init.lua's <leader>k stages the canned block
+# context.txt lines. Mechanism: init.lua's <leader>k stages the canned block
 # above the current article's !!! floor. Witness: the \k-staged block rode
 # a Prompt section in the same compile whose receipt still showed PENDING —
 # flipped by patch, never by drift, one unwitnessed turn exactly as allowed.
@@ -396,7 +396,7 @@ AI_PHOOEY_CHOP = r"""#
 # citation address space, requires freezing the corpus first. Immutability is not
 # the apparatus's enemy; it is its precondition. Footnote = an out-of-band,
 # non-destructive annotation channel (the OOB EDIT RULE in print). Margin = the
-# user-writable per-copy layer (adhoc.txt is this book's margin). Errata/addenda =
+# user-writable per-copy layer (context.txt is this book's margin). Errata/addenda =
 # append-only patches that preserve the original error ON PURPOSE, so the delta
 # stays visible.

@@ -616,11 +616,11 @@ AI_PHOOEY_CHOP = r"""#
 # mission. The artifact travels as a legible, open-standard text ledger
 # (resembling AGENTS.md or SKILL.md structures), presenting zero friction to
 # the receiver while packing a complete, verifier-enforced cognitive
-# environment under the skin. Concrete command-surface witness: `ahc` remains
-# the explicit ADHOC_CHOP author/workbench compiler, while `cpr` resolves the
-# walk router and invokes the same prompt_foo.py machinery through WALK_CHOP
-# with a lean frame, no tree, quiet output, and invocation-local
-# PIPULATE_ADHOC_FILE. The user names the goal; the framework stays behind it.
+# environment under the skin. Concrete command-surface witness: `compile`
+# (ahc) is the workbench compiler, the walk writes the same context.txt that
+# `compile` reads, and the reading frame is one flag on the same command,
+# compile --frame lean --chop WALK_CHOP (2026-09-25). The user names the
+# goal; the framework stays behind it.

 # THE MOTHER CAT KATA (MCK) (earmark-seeded 2026-07-28): the record-and-playback
 # session that carries the human by the scruff -- bookmark to bookmark, a popped
@@ -1332,7 +1332,7 @@ foo_files.py   # <-- THIS file. Content compiler router. Makes it very meta.

```
 # LAUNCH
 # walk                    # <-- Repo-root wrapper; delegates to mck.sh, NOT to scripts/walk.py
 # assets/installer/mck.sh     # <-- THE LAUNCHER (curl -fsSL pipulate.com/mck.sh | bash): finds the workshop, forces the spoken rehearsal on first contact, asks for the word RIDE
-# scripts/mother_cat.py       # <-- THE RIDER (alias: mothercat), Car B: actuates walk.py's validated plan; per-run capture banking, DECANT, the adhocwalk.txt writer
+# scripts/mother_cat.py       # <-- THE RIDER (alias: mothercat), Car B: actuates walk.py's validated plan; per-run capture banking, DECANT, the context.txt writer
 # scripts/boot_menu.py        # <-- The one-door command list; short on shell entry, expanded by `all`, both rendered from tuples in the file
 # scripts/sources_menu.py     # <-- The roster `conn` prints; filename stays descriptive, typed word is shorter
 #
@@ -1354,7 +1354,7 @@ foo_files.py      #  <-- THIS file. Content compiler router. Makes it very meta.
 # imports/voice_synthesis.py  # <-- Piper: scripted narration, never a model (ATTRIBUTED-VOICE)
 #
 # HAND OFF
-# prompt_foo.py             # <-- Compiles the selected router (adhocwalk.txt: `epr` edits it, `cpr` compiles it) into a payload
+# prompt_foo.py             # <-- Compiles context.txt (`context` opens it, `compile` builds it) into a payload
 # scripts/foo_cartridge.py    # <-- The second seal, over the EVIDENCE: payload.md, prompt.md, manifest.json
 # scripts/foo_replay.py       # <-- Context extraction and attention checks; not browser or API replay
 # tests/test_mck_rep2.py      # <-- Rep 2: the earmark's owed side-by-side witness
@@ -1365,7 +1365,7 @@ foo_files.py      #  <-- THIS file. Content compiler router. Makes it very meta.
 # assets/trails/botify_pageworkers.yaml  # profile botify; SETTLE real; UNRIDDEN -- the only trail on a non-default profile, and no such ride has ever been witnessed
 # assets/trails/jira_for_you.yaml       # profile default; SETTLE real (Atlassian sign-in); top of the ticket loop; RIDDEN 2026-09-01 and DECANTED, the sign-in moment unrecorded (a cold profile records it)
-# assets/trails/ticket.yaml             # profile default; SETTLE real; url_env x2 (JIRA, BOTIFY) fed by one gitignored exports file per ticket under Notebooks/Client_Work/tickets/; stop two of the ticket loop; RIDDEN 2026-09-01 to stop 1 then REFUSED at stop 2 on an unset url_env, Linux and Mac -- the ride that bought the rider its PRE-FLIGHT. RULING: the connector does NOT run at DECANT (DECANT gates only CAPTURE-fenced material; API auth is a different wallet; the rider never harvests). The bridge to the issue text is a ! line in adhoc.txt.
+# assets/trails/ticket.yaml             # profile default; SETTLE real; url_env x2 (JIRA, BOTIFY) fed by one gitignored exports file per ticket under Notebooks/Client_Work/tickets/; stop two of the ticket loop; RIDDEN 2026-09-01 to stop 1 then REFUSED at stop 2 on an unset url_env, Linux and Mac -- the ride that bought the rider its PRE-FLIGHT. RULING: the connector does NOT run at DECANT (DECANT gates only CAPTURE-fenced material; API auth is a different
```

wallet; the rider never harvests). The bridge to the issue text is a ! line in context.txt.
 # assets/trails/se_ticket.yaml   # profile default; SETTLE real; the SE ticket template:
for_you (literal) + issue (JIRA, required) + project/slack/confluence (optional, skipped when unset); RIDDEN 2026-09-02, 3 of 5 captured, 2 skipped, 72,383 B decanted, card showed both rows; harvest regexes on optional stops are PLACEHOLDERS (botify_analysis's already fullmatched a live analysisSlug); guidance strings hand-count "of five" and go stale on the sixth stop
 #
 # UNLINKED PAGES (npvg.org; nothing links to them and no page links out, so a trail is the only edge; nixops.sh rsyncs them, no rebuild)
@@ -1383,23 +1383,24 @@ foo_files.py   # <-- THIS file. Content compiler router. Makes it very meta.
 # network body. Preview: DECANT releases selected, capped text, never the raw
 # archive. Disclosure: captures.disclosed.json is a separately prepared
 # derivative with omissions and review status; nothing writes it for you.
-# Selection: adhocwalk.txt names the checked preview and, commented beneath it, the raw captures.md;
+# Selection: context.txt names the checked preview and, commented beneath it, the raw captures.md;
 # the list is neither disclosure approval nor the evidence.
 #
-# THE HANDOFF (97feb327; amended 2026-09-16): a complete, nonempty run has the
-# rider write adhocwalk.txt WHOLE beside the selected router, 0600,
-# os.replace'd -- the checked preview as the one uncommented line once DECANT
-# released it, the whole archive as a commented line beneath, and the archive
-# alone, labelled, when no preview was saved. IT REPLACES, IT NEVER MERGES, and
-# because `epr` now invites a hand-typed `!` line into this very file, it
-# REPLACES ONLY A FILE IT COULD HAVE WRITTEN: one uncommented absolute path to
-# an archive or a preview and nothing else. Any other uncommented line means a
-# human owns the file; the rider prints the new path lines and touches nothing,
-# and deleting the file lets the next walk rewrite it. Riding from a shell where
-# PIPULATE_ADHOC_FILE already names adhocwalk.txt refuses by name, non-fatally,
-# archive preserved, which is why cpr assigns in a subshell rather than
-# exporting. SELECTING IS NOT INCLUDING: naming adhocwalk.txt inside another
-# list includes its text and never recurses into its entries.
+# THE HANDOFF (97feb327; amended 2026-09-16 and 2026-09-25): a complete,
+# nonempty run has the rider write context.txt WHOLE -- the one file
+# `context` opens and `compile` reads, PIPULATE_ADHOC_FILE or the repo root
+# -- 0600, os.replace'd: the checked preview as the one uncommented line once
+# DECANT released it, the whole archive as a commented line beneath, and the
+# archive alone, labelled, when no preview was saved. IT REPLACES, IT NEVER
+# MERGES, and because `context` invites a hand-typed line into this very
+# file, it REPLACES ONLY A FILE IT COULD HAVE WRITTEN: introduction.md plus
+# one uncommented absolute path to an archive or a preview and nothing else.
+# Any other uncommented line means a human owns the file; the rider prints
+# the new path lines and touches nothing, and deleting the file lets the next
+# walk rewrite it (witnessed 2026-09-25 against the operator's own working
+# list: two refusals, two paths printed, nothing written). SELECTING IS NOT

+# INCLUDING: naming context.txt inside another list includes its text and
+# never recurses into its entries.
 #
 # AUTH RULING (banked 2026-08-09, source-witnessed): THERE IS NO AUTH FIELD IN
 # THE TRAIL SCHEMA. walk.py enforces set-difference in BOTH directions over the
@@ -1714,7 +1715,7 @@ foo_files.py   # <-- THIS file. Content compiler router. Makes it very meta.
 #
 # THE WORDS (census of alias and function definitions in flake.nix, deed 1469,
 # corrected at deed 1470 by the flake itself): after the walk a newcomer's
-# whole vocabulary is walk, plan, voice, epr, cpr and sources; warm and
+# whole vocabulary is walk, plan, voice, context, prompt, compile and conn; warm and
 # weblogin settle a login before a ?URL; bot, mcp and webclip are the
 # operator's. THE SEVEN CONNECTOR WORDS (jira, gmail with email as a second
 # spelling, confluence, gsc, sheets, slack, botify) are the THIRD TIER, not
@@ -2273,7 +2274,7 @@ MATCHBOOK_CHOP = r"""
 # § THE EPITAPH COUNTER (banked 2026-08-28) -- a removal probe anchored on the removed string cannot read zero when the patch quotes that string in a comment; the tombstone is a hit. Anchor on syntax only mechanism produces, print lines rather than counts, or predict the epitaph: "1, and it will be the comment" is correct and "0" is not.
 # § THE SURVIVING FALLBACK (banked 2026-09-04) -- sibling of THE EPITAPH COUNTER, opposite direction: a probe anchored on a string the patch DEMOTES rather than DELETES reads the same number before and after, because the survivor is the point of the change. Conviction: grep -c "get('url'" predicted 1 -> 0 against a patch that rewrote `.get('url', default)` into `base_url or url or default`, so the key it counted was deliberately kept. Ask before predicting a zero: does the patch remove this string, or demote it? If demote, the probe is unfalsifiable and the git diff is the only honest witness. RETIRE such a probe rather than repairing it; a reading that is identical in both worlds trains the operator to skip probe output. GIT FORM (2026-09-05): git log -S counts a string per commit and is blind to a commit that MOVES it, so -S 'pipulate/core.py' named an old refactor while 3799ffbc carried that path from a chapter into the paintbox; -G matches diff lines and would have named it. A moved string is a demoted string.
§ THE CENSUS IS NOT A STRADDLE (banked 2026-09-04) -- some readings answer "is this branch reachable at all", not "did my patch land". They read IDENTICALLY before and after BY DESIGN, and their number is a RULING that decides whether a car is owed. Label them CENSUS in the probe prose so an unchanged output is never mistaken for a failed patch, and never predict a delta for one. Conviction: 0 of 1446 and 0 of 126 posts lack a frontmatter permalink, which proved two disagreeing fallbacks are both unreachable and cancelled the car that would have picked a winner between them.
-# § THE MULTI-LINE PROBE CANNOT BE ECHOED (banked 2026-09-04) -- the ! executor spawns a NON-interactive sh -c, so a probe spanning newlines inside quotes runs clean by hand and dies at "unexpected EOF" in the compile lane; adhoc.txt then reads its continuation lines as separate paths and warns about de-prefixed commands. THE PROBE ECHO INVARIANT therefore demands ONE LINE per probe: write it as a single semicolon-joined -c string, or as a real file invoked by path. A probe that cannot survive its own echo is not echoable, and the operator lane becomes its only witness.
+# § THE MULTI-LINE PROBE CANNOT BE ECHOED (banked 2026-09-04) -- the ! executor spawns a NON-interactive sh -c, so a probe spanning newlines inside quotes runs clean by

hand and dies at "unexpected EOF" in the compile lane; context.txt then reads its continuation lines as separate paths and warns about de-prefixed commands. THE PROBE ECHO INVARIANT therefore demands ONE LINE per probe: write it as a single semicolon-joined -c string, or as a real file invoked by path. A probe that cannot survive its own echo is not echoable, and the operator lane becomes its only witness.
 # § THE SUBSET REPLACE (banked 2026-09-04) -- a REPLACE block that is a substring of its SEARCH block (every dedupe car, by construction) is already in the file BEFORE the patch, so an already-applied heuristic reads a plain not-found as "ambiguous (found N times)" and hides the nearest-line diagnostic; apply.py now skips that heuristic when the REPLACE sits inside the SEARCH, and a dedupe car cuts the second copy by gated line range, never by a SEARCH spanning the seam between copies, because the seam is where the blank line lives. Sibling of THE EPITAPH COUNTER (a probe finds the patch's text) and THE CONTIGUITY COROLLARY (the seam the author cannot see). A range bounded by the NEXT HEADER minus one contains the separator blank by construction, so the gate trims one trailing blank before it counts, or its GO is unreachable and the gate is a refusal-only witness in a gate's costume; ask what the gate prints when the file is exactly as believed. Witnessed 2026-09-04: three GOs, the trim firing once and idle twice.
 # THE SOURCE-OF-SOURCE SECRET SCRUB RULE (banked 2026-08-28,
 # receipt-witnessed): move credential EPHEMERA upstream only when the match is
@@ -2472,7 +2473,7 @@ MATCHBOOK_CHOP = r"""
 # - EARMARK: THE SELF-MATCHING PATTERN (banked 2026-08-05, twice-convicted in one transcript): pkill -f and pgrep -f match the FULL COMMAND LINE, and the shell running the script carries the pattern in its own argv -- so an unescaped pattern kills or counts the process that issued it. CONVICTION: flake.nix's publish() [4/4] block protected two patterns with the [.] trick (a regex demanding a literal dot the argv literal does not contain) and MISSED THE THIRD, so every `publish --reboot` killed its own ssh session at that line. The guard must be applied to EVERY pattern in a block, not most of them -- a partially-guarded block READS as guarded. WITNESS: after the fix, `publish force --reboot` printed `new_count=1`, `new=534841`, and the watchdog verdict line, three lines that had never once appeared in any prior transcript.
 # - EARMARK: THE VERIFIER THAT NEVER RAN (banked 2026-08-05, same conviction): a verification block placed AFTER the action it verifies is dead code if the action can kill the reporter, and its silence is indistinguishable from success. flake.nix's [4/4] block held `sleep 12`, a pgrep re-count, and three verdict branches -- none had EVER executed, while a green Atomic Deployment Complete printed underneath every time. THE DISCRIMINATION QUESTION applied to a MISSING line: what does this print in the world where the verifier died? The same checkmark. STANDING CONSEQUENCE: when a receipt block has an expected line that is ABSENT, treat the absence as the finding; do not read the surviving lines as the whole receipt. Third shape in the family -- REFUSAL-ONLY WITNESS is a branch never observed, THE SUCCESS-ONLY WITNESS is a failure never reportable, this is a witness never REACHED.
 # § INCOMMENSURABLE MEASUREMENTS (banked 2026-08-05) -- a probe can return a CORRECT number about a DIFFERENT question, and the receipt looks authoritative because the number is right; name the exact expression the mechanism evaluates, then confirm the probe evaluates THAT expression and not a plausible neighbour. Cousin of THE DISCRIMINATION QUESTION's family: it sees something real and irrelevant.
-# - EARMARK: THE STALE OVERLAY (banked 2026-08-05, Manifest-arbitrated): the adhoc.txt pasted into an article and the adhoc.txt the compiler READ can disagree within a single turn, and only the Manifest's LIVE COMMAND RECEIPTS can tell them apart. CONVICTION: a turn pasted three new probe lines while the compile executed the five from two compiles earlier --

the pasted context described a file the compiler never opened. STANDING CONSEQUENCE: the Manifest is the sole authority on what ran; when the pasted overlay and the receipts disagree, rule from the receipts and say so out loud. The map can outrun the territory inside one turn.
+# - EARMARK: THE STALE OVERLAY (banked 2026-08-05, Manifest-arbitrated): the context.txt pasted into an article and the context.txt the compiler READ can disagree within a single turn, and only the Manifest's LIVE COMMAND RECEIPTS can tell them apart. CONVICTION: a turn pasted three new probe lines while the compile executed the five from two compiles earlier -- the pasted context described a file the compiler never opened. STANDING CONSEQUENCE: the Manifest is the sole authority on what ran; when the pasted overlay and the receipts disagree, rule from the receipts and say so out loud. The map can outrun the territory inside one turn.
 # - TODO (banked 2026-08-05, receipt-scoped): SEVEN in-window articles are the entire residual on-air bracket noise -- 2026-07-31-absolute-timing-and-side-channel-sanitization,
2026-08-01-agentic-web-content-negotiation-telemetry,
2026-08-01-hatching-the-seed-installer-automation,
2026-08-01-mother-cat-kata-deterministic-walk,
2026-08-03-debugging-self-referential-prompt-guards,
2026-08-04-disposable-installations-durable-backups,
2026-08-05-sentinel-splices-workspace-tree-automation. They age out of the 7-day window unaided, so this backlog SHRINKS without action. HAND-EDIT ONLY: several quote the patch-protocol markers verbatim, which is a DELIMITER COLLISION under the OOB EDIT RULE -- vim, never apply.py. Full 300-article list is 20 and lives in the 2026-08-05 compile receipt.
 # - TODO (banked 2026-08-05, publish-transcript convicted): nixops.sh's rsync --delete ships the Z640's __pycache__ (cpython-312) to Honeybot and DELETES Honeybot's own (cpython-313) on every publish -- six lines of churn per deploy for zero benefit, since a 3.12 pyc can never load on 3.13. Add --exclude=__pycache__. Separately, trimnoir tracks scripts/aquarium.py and scripts/aquarium_tui.py and nothing else (receipt: git ls-files scripts/ returns exactly those two), so the post-receive checkout restores them and the rsync deletes them again, forever. Delete them from trimnoir. TWO WRITERS, ONE DIRECTORY -- cosmetic today only because content_loader.py is NOT tracked there, and a live regression path the day it is.
 # - TODO (banked 2026-08-05, two-sided receipt): stream.py's stdout is TTY-ONLY and structurally unreadable as text. Receipt: journalctl over 45 minutes returned journal_lines=761 (the journal is alive) and unbalanced=0 (the warnings are not in it), and configuration.nix defines `stream` as a writeShellScriptBin watchdog in environment.systemPackages with NO systemd unit -- so it runs in a terminal in the XFCE session and its output exists only as pixels for OBS. THE OPERATOR HAS BEEN DIAGNOSING FROM A VIDEO FEED for the whole census ride. Cheapest capture: tee the watchdog's python3 invocation to a file inside the `stream` script. That is a configuration.nix plus nixos-rebuild ride of its own.
(nix) pipulate $ m
📝 Committing: chore: Update probe echo rule to use context.txt for echoing probes
[main 8b4072b6] chore: Update probe echo rule to use context.txt for echoing probes
 1 file changed, 30 insertions(+), 29 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 44, done.
Counting objects: 100% (44/44), done.

Delta compression using up to 48 threads
Compressing objects: 100% (31/31), done.
Writing objects: 100% (32/32), 4.83 KiB | 705.00 KiB/s, done.
Total 32 (delta 22), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (22/22), completed with 11 local objects.
To github.com:pipulate/pipulate.git
   7cb51ba2..8b4072b6 main -> main
(nix) pipulate $

Bundling Browser Signals into Reproducible Assets

Wow, Fable 5.1 is continually impressive. Nix shell rebuild and reload of init.lua here in NeoVim.
(nix) pipulate $ exit
exit
(sys) pipulate $ ndq
(nix) pipulate $ context
(nix) pipulate $

Hmmm. The file is still named adhoc.txt. I've got some looking over things to do. But maybe the priority was to have the walk write into this file now properly. I know we're creating colliding context issues as we work on the system that writes the very file we're using to write this article. A walk while writing an article could... Hmmm, we can always just append to the bottom. Things can be additive without interfering. Someone doing two parallel things that use the adhoc.txt or context.txt file differently are big boys and girls and can deal with commenting appended-to-the-bottom things in and out accordingly. It might be nice to just take walks whenever you want and have that stuff append at the bottom as if building the next context with a walk. That's completely valid.
4: Prompt: THE AFTER READING for the leftovers ride, deltas over absolutes, one probe at a time:

1. The adhoc census: prompt_foo.py 7 -> absent, flake.nix 6 -> absent, foo_files.py 18 -> 6 (the six survivors are dated: the 2026-09-21 receipt, the 2026-09-19 and 2026-09-16 TODOs, the DE-HARDWIRE TODO, the ahc-toggle TODO, the 2026-07-17 pin; a forget ride rules on them, not a rename), scripts/git_hooks/pre-commit 3 -> 1 (the for-loop line names all three), scripts/walk.py 1 -> absent, .gitignore 2 by design. Unchanged and next on the list: connectors (README plus seven files), GLOSSARY.md 3, the two SKILL.md files, scripts/two_arm.py, and imports/ascii_displays.py at 7, which is sealed art and needs a reseal, not a sed.
2. The export hunt: one file:line naming PIPULATE_ADHOC_FILE, or another empty set, in which case the next probe is env -0 | tr '\0' '\n' | rg ADHOC plus rg -l PIPULATE_ADHOC_FILE /etc /run/current-system/etc 2>/dev/null. The rename car (mv adhoc.txt to context.txt, rm adhocwalk.txt, edit the export, one gated paste, then exit and ndq) rides only once this line names the file.
3. The hook straddle: scripts copy 0 -> 3, .git/hooks copy 0 -> 3 after ignition; 3 and 0 means the shell was not re-entered.
4. The epr/cpr census: introduction.md 4 -> absent, prompt_foo.py 4 -> 1 (the 2026-09-18 conviction line), foo_files.py -3, scripts/mother_cat.py -1, scripts/boot_menu.py 1 both

taps (the epitaph). Any hit under assets/installer or remotes/honeybot/www/npvg.org is a car for next turn, and the pad's is a nixops.sh truck, not a push.
5. The adhoc word count: scripts/walk.py 2 -> 1 (the adhoc_mutation JSON key stays until the ADHOC-name ruling), init.lua 2 -> 0, scripts/mother_cat.py 1 -> 0.
6. door 2: two lines -> door2=0.
7. The generated hook text: 0 -> 1 without ignition, because nix eval reads the dirty worktree; the function in your shell still needs exit then ndq.
8. The nvim gate: green both taps; a gate, not a landing.
9. The sound census: two .wav files under ~/.local/share/pipulate, the reader's path line in tools/scraper_tools.py, and .gitignore's *.wav line. Ruled: they move into assets/sounds/ with a README naming OpenGameArt 16323 and Freesound 611113 as CC0, .gitignore gets !assets/sounds/*.wav below its *.wav line, and the reader prefers the repo path with ~/.local/share as the fallback so the Mac stops being silent. Car 1's receipt: HEADER_PRUNED with template=40 and tail_muted near 22 (13 files and 9 probe echoes), or HEADER_ALREADY_PRUNED if you had already cut it by hand.

NEXT TURN'S CARS, once the AFTER is in: the rename (probe 2 names the file), the sounds (probe 9, with tools/scraper_tools.py in context), and a ruling before any car on the walk writing into a human's file: the guard that refused today exists so a walk never overwrites your list, so the shape that keeps it is an append under a dated "# --- WALK ---" marker with the vim lesson only when the file is absent, and a replace never; say yes or no and mother_cat.py is in context for it. RULINGS CARRIED, not cars: whether the reading frame gets a word back or stays compile --frame lean; whether PIPULATE_ADHOC_FILE, ADHOC_CHOP, the ADHOC SLOT markers and walk.py's adhoc_mutation key are renamed, each a gated car with its readers in context; whether page 3 of the walk names prompt and compile beside its question; and the GLOSSARY.md purge, which is the operation's largest remaining bite and a ride of its own.
5: Deliverables: Your priority is to finish what you've started here and confirm it's all working as you planned. With any spare tokens you can think about the white rabbit ASCII art under operation stick bug targeting people who are feeling that itch and don't know it and code auditors who want to kill it because they don't understand it.
Note: Okay this is 6 minutes and 56 seconds worth of thought. I guess things must be getting easier.
Reviewing the after-change receipts for the compile results.

Locating the environment file behind the pending rename.

Recounting file matches after the hook and rename changes.

Tracing extra file matches to an overlooked paragraph.

Verifying each file's match count against predicted deltas.

Designing a safer gate condition for the header rename.

Sequencing the shell exit and reload to prove the source.

Moving courtesy sound files into the repository with fallback logic.

Designing append-versus-replace logic for walk file updates.

Drafting the appended-block format and marker structure for edits.

Composing status messages for each write mode variant.

Splitting one large edit into smaller anchored search blocks.

Redefining writer-owned files to protect commented-out lines.

Verifying replace-versus-append behavior across walk scenarios.

Writing the atomic file swap and marker-block replacement logic.

Updating caller messages to report the new write outcomes.

Drafting a bounded probe to test both write modes.

Assembling a one-line probe to compare write modes.

Scripting the environment-file rename with safety checks.

Deciding whether to strip the header's ad hoc notes and art.

Ordering the prune and rename steps to avoid breakage.

Ordering the sound-directory fallback for a repo-first override.

Ignoring the .wav ignore rule for shipped sound files.

Verifying overridden fallback paths under a controlled home directory.

Confirming file-tracking status with a reliable git command.

Rewriting the checksummed art's caption to fit the new framing.

Preparing the audit review sections and the art draft.

Splitting the sound-file commit into two separate car sequences.

Rewriting the router's ownership rules for appended edits.

Implementing atomic append logic for a human's existing file.

Rewriting the marker block to preserve edits above it.

Anchoring the write-block edit to unique surrounding lines.

Updating callers and message text for the new receipt values.

Shortening the router comment to fit the line limit.

Drafting a script to prune outdated header notes.

Predicting template length and muted-line counts for the header prune.

Sequencing steps and drafting probes for the next work session.

Drafting bounded probes to confirm ownership markers across files.

Verifying prerequisite files exist before copying sound assets.

Composing the ASCII art draft with its verification caption.

Sequencing the multi-car turn and drafting the ignition instructions.

Cross-checking search text against the exact payload block.

Locating precise anchor lines for the atomic write block.

Fable 5.1 Extra: Context verified. The cars below patch scripts/mother_cat.py, tools/scraper_tools.py and .gitignore, all in this payload, plus two files outside the repo that the pastes name by path. Two files next turn needs are absent and ride in NEXT CONTEXT: foo_files.py (its HANDOFF paragraph says "IT REPLACES, IT NEVER MERGES," which Car 5 makes false for a human's file) and imports/ascii_displays.py (the bunny's home, sealed with a CRC).

The AFTER reading

1. Census: prompt_foo.py 7→absent, flake.nix 6→absent, init.lua 2→absent, scripts/walk.py 1→absent, foo_files.py 18→6, scripts/git_hooks/pre-commit 3→1, .gitignore 2. Every delta landed.
2. The export is found, and it is doubled: .env lines 12 and 13 carry the same PIPULATE_ADHOC_FILE= assignment, and the flake sources that file on every entry. The rename rides now.
3. Hook straddle 0→1 on both copies. I predicted 3 and was counting the phrase "context file," which the regex does not match; the one matching line is the for loop, and the two copies agreeing is the ignition witness.
4. Retired words: introduction.md 3→absent (lines, not words: my 4 was occurrences); prompt_foo.py 5→2, delta −3 as predicted with the absolutes off by one, so one more cpr line survives than I named; foo_files.py 15→10, delta −5 against a predicted −3, because the HANDOFF paragraph carried two more lines I rewrote and did not count; mother_cat.py 1→1, because the TRAINING WHEELS line kept "Mac epr screen" as a dated fact, so the probe was anchored on a string the patch demoted and your diff is the only witness; boot_menu.py 2, the epitaph, two lines not one.
5. walk.py 2→1, init.lua 2→0, mother_cat.py 1→0. 6. door 2 gone. 7. The hook text reads 1.

    8. Gate green.
6. Sounds: two files, 645,230 and 444,494 bytes; the loader reads one directory under your home; .gitignore:13 ignores every .wav.

Car 1 refused correctly. You had already edited the first line of the file, and the gate was keyed on that line; THE OPERATOR IS A VARIABLE. The retry below keys on the template's own first line instead, so it is idempotent by content and not by history.

On the ruling: yes, the walk appends. Car 5 makes a human's file append-only, one marked block per walk at the bottom with the preview active and the archive commented, and a second write in the same ride (after DECANT) replaces only its own block. A file the writer made is still replaced whole, so a newcomer's file never piles up.

The bunny, for the next turn

The drawing stays; the words change. Same seven lines, so the seal is one CRC away, and registering it is the three-region edit in imports/ascii_displays.py with figurate white_rabbit reading Drift: 0 as its ignition. A draft, columns to be set at registration:

```
                ( "The AI said so" is a transcript, not a record. )
                                      O       /)  ____      This drawing is checksummed.
>  Same files, same words, same bytes:      o /)\__//  /    \    If it reaches you altered, the
>  a change you can replay is a change     ___(/_ 0 0  |      |   model rewrote what it was told to
>  you can put your name behind.         *(    ==(_T_)== NPvg |   copy, and that is the finding.
>  One diff per turn, a reading before    \  )   ""\  |      |   Verify with nothing installed:
>  and after, and a digest you can check.  |__>-\_>_>  \____/    python scripts/foo_cartridge.py
foo.zip
```

It answers the reviewer's first question before it is asked (a cockpit voice recorder is not a flight data recorder) and hands over the instrument on the last line, which are the two of your three that a card can carry.

1. PROBES

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter.

```
rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate ~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc /etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
ls -1 ~/.local/state/pipulate/
HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file; print("SOUNDS", _sound_file("tick.wav"), _sound_file("ding.wav"))' 2>&1 | tail -n 1
git ls-files assets/sounds | wc -l
PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib; sys.path.insert(0,"scripts"); import mother_cat as m; p=pathlib.Path("/tmp/ctxprobe.txt"); p.write_text("# mine\nflake.nix\n"); a=pathlib.Path("/tmp/walk-probe/captures.md"); r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d mine=%d lines=%d" % (t.count("WALK walk-probe START"), t.count("flake.nix"), len(t.splitlines())))' 2>&1 | tail -n 1
```

for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER MERGES' "$f" || echo 0)"; done
rg -nw cpr prompt_foo.py

Line 1 is a CENSUS this turn; no tracked line changes. Line 2 straddles the rename: one .env line and a new ENV= after ignition. Line 3 straddles the state directory. Line 4 is the Mac's world: with no home copy, the loader finds nothing before and the repo copies after. Line 5 is 0→3 once the files are committed. Line 6 is a fixture probe: it writes a two-line file under /tmp and calls the writer twice; before, it ends in the refusal, after, it prints one block, your line intact, and the line count. Line 7 is the census for next turn's foo_files.py car. Line 8 names the two cpr survivors so the next turn can rule on them.

2. NEXT CONTEXT

```
# --- NEXT TURN'S CARS: the HANDOFF paragraph, the bunny, the cpr survivors ---
foo_files.py
imports/ascii_displays.py
prompt_foo.py
# scripts/mother_cat.py    # <-- only if Car 5 refused; line 6 is its witness
# tools/scraper_tools.py   # <-- only if Car 3 refused; line 4 is its witness
# .gitignore
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate ~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc /etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
! ls -1 ~/.local/state/pipulate/
! HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file; print("SOUNDS", _sound_file("tick.wav"), _sound_file("ding.wav"))' 2>&1 | tail -n 1
! git ls-files assets/sounds | wc -l
! PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib; sys.path.insert(0,"scripts"); import mother_cat as m; p=pathlib.Path("/tmp/ctxprobe.txt"); p.write_text("# mine\nflake.nix\n"); a=pathlib.Path("/tmp/walk-probe/captures.md"); r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d mine=%d lines=%d" % (t.count("WALK walk-probe START"), t.count("flake.nix"), len(t.splitlines())))' 2>&1 | tail -n 1
! for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER MERGES' "$f" || echo 0)"; done
! rg -nw cpr prompt_foo.py
```

3. PATCHES

Five cars, in this order. Cars 1 and 2 are terminal pastes that change files outside the repo, so they have no d and no m; each prints its own receipt. Car 3 is a patch, Car 4 is a paste that lands two files for m to pick up, Car 5 is a patch.

Car 1 (terminal paste): the live file's header becomes the plain template

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter.

```
python - <<'EOF'
import os, sys
from pathlib import Path
root = Path(os.environ.get("PIPULATE_ROOT", "."))
sel = os.environ.get("PIPULATE_ADHOC_FILE", "")
p = Path(sel).expanduser() if sel.strip() else root / "context.txt"
if not p.is_file():
    sys.exit(f"STOP: no context file at {p}")
ff = (root / "foo_files.py").read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(ff) if l.rstrip() == "# --- START `context.txt` TEMPLATE ---"]
E = [i for i, l in enumerate(ff) if l.rstrip() == "# --- END `context.txt` TEMPLATE ---"]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: template markers", S, E)
template = ff[S[0] + 1:E[0]]
while template and not template[-1].strip():
    template.pop()
lines = p.read_text(encoding="utf-8").split("\n")
if lines[:1] == template[:1]:
    print(f"HEADER_ALREADY_PRUNED {p}; nothing written")
    sys.exit(0)
C = [i for i, l in enumerate(lines) if l.startswith("# Context 1 (Edit-in")]
assert len(C) == 1, ("STOP: Context 1 marker", C)
tail, muted = [], 0
for l in lines[C[0] + 1:]:
    if l.strip() and not l.lstrip().startswith("#"):
        l, muted = "# " + l, muted + 1
    tail.append(l)
out = template + ["", "# Context 1"] + tail
p.write_text("\n".join(out), encoding="utf-8")
print(f"HEADER_PRUNED {p} header_lines_replaced={C[0] + 1} template={len(template)}
tail_kept={len(tail)} tail_muted={muted}")
EOF
```

Car 2 (terminal paste): the file takes its new name, and .env says so once

Run Car 1 first. Then copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter. It rewrites only the PIPULATE_ADHOC_FILE lines of .env and prints counts, never the file.

```
python - <<'EOF'
import os, sys
from pathlib import Path
```

```
root = Path(os.environ.get("PIPULATE_ROOT", "."))
env = root / ".env"
old, new = "adhoc.txt", "context.txt"
state = Path.home() / ".local/state/pipulate"
src, dst, walk = state / old, state / new, state / "adhocwalk.txt"
lines = env.read_text(encoding="utf-8").split("\n")
hits = [i for i, l in enumerate(lines) if l.startswith("PIPULATE_ADHOC_FILE=")]
bad = [i for i in hits if old not in lines[i]]
assert hits and not bad, ("STOP: .env export lines", len(hits), len(bad))
assert src.is_file() and not dst.exists(), ("STOP: state files", src.is_file(), dst.exists())
keep = lines[hits[0]].replace(old, new)
out = [l for i, l in enumerate(lines) if i not in hits[1:]]
out[hits[0]] = keep
env.write_text("\n".join(out), encoding="utf-8")
src.rename(dst)
removed = walk.exists()
if removed:
    walk.unlink()
print(f"RENAMED {src.name}->{dst.name} env_lines_before={len(hits)} env_lines_after=1
adhocwalk_removed={removed}")
EOF
```

Car 3 (patch): the walk sounds have a home in the repo

Target: assets/sounds/README.md
[[[WRITE_FILE]]]
```
# Walk sounds

Two short sounds the guided walk plays while a page opens. `tick.wav` loops
from the moment the browser is launched until the page has loaded, and
`ding.wav` plays once when it has. Both are CC0 (public domain dedication)
and were re-encoded to 16-bit PCM, 44.1 kHz, stereo with ffmpeg on
2026-09-19.

- `tick.wav` (3.66 s): "clock-1.wav", OpenGameArt node 16323, by AntumDeluge, CC0.
- `ding.wav` (2.52 s): Freesound sound 611113 by 5ro4, CC0.

A copy of either file under `~/.local/share/pipulate/` is played instead of
the one here, so you can swap in your own without touching the repo. No file
in either place means no sound.
```
[[[END_WRITE_FILE]]]
Target: .gitignore
[[[SEARCH]]]
```
*.wav
*.tsv
```
[[[DIVIDER]]]
```
*.wav
```

```
# the walk's two CC0 sounds ship in the repo; assets/sounds/README.md names their sources
!assets/sounds/*.wav
*.tsv
[[[REPLACE]]]
Target: tools/scraper_tools.py
[[[SEARCH]]]
# --- Optional audio during browser startup ---
# Operator-owned courtesy sounds live outside the repo. tick.wav loops while
# the browser is opening; ding.wav plays once when the page-load phase ends.
# No file means no sound. Linux uses aplay; Darwin uses afplay.
SCRAPE_MUSIC_MARKER = "pipulate-scrape-music"
SOUND_DIR = Path.home() / ".local/share/pipulate"
[[[DIVIDER]]]
# --- Optional audio during browser startup ---
# tick.wav loops while the browser is opening; ding.wav plays once when the
# page-load phase ends. Both are CC0 and ship in assets/sounds/ (its README
# names the sources) since 2026-09-26, after a week in which the Mac was
# silent because the only copies lived under one home directory. A copy
# under ~/.local/share/pipulate still wins, so an operator can swap in their
# own. No file in either place means no sound. Linux uses aplay; Darwin
# uses afplay.
SCRAPE_MUSIC_MARKER = "pipulate-scrape-music"
SOUND_DIRS = (
    Path.home() / ".local/share/pipulate",
    Path(__file__).resolve().parent.parent / "assets" / "sounds",
)
[[[REPLACE]]]
Target: tools/scraper_tools.py
[[[SEARCH]]]
def _sound_file(name):
    path = SOUND_DIR / name
    return path if path.is_file() else None
[[[DIVIDER]]]
def _sound_file(name):
    for sound_dir in SOUND_DIRS:
        path = sound_dir / name
        if path.is_file():
            return path
    return None
[[[REPLACE]]]
```

Car 4 (terminal paste): the two files land

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter, then d, m.

```
cp ~/.local/share/pipulate/tick.wav ~/.local/share/pipulate/ding.wav assets/sounds/ && ls -l assets/sounds/
```

Car 5 (patch): a human's file is appended to, never replaced

Target: scripts/mother_cat.py
[[[SEARCH]]]
    "# The next walk rewrites this file (shorter notes) unless you have edited it.\n"
[[[DIVIDER]]]
    "# Walks rewrite this file until you edit it; after that they append below.\n"
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
wheels are off. A human who edits the router keeps it (the guard in the
writer); one who deletes it gets the lesson back on the next walk.
[[[DIVIDER]]]
wheels are off. A human who edits the router keeps it: later walks append
a marked block below their lines; deleting it brings the lesson back.
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
    """Replace one local selection; never read, disclose or compile its evidence.
[[[DIVIDER]]]
    """Write one local selection; never read, disclose or compile its evidence.
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
    THE ROUTER IS THE NEWCOMER'S FILE (2026-09-16): `context` opens it, so it is
    replaced ONLY while it is a file this writer could have produced -- the
    introduction line at most, plus one uncommented absolute path to an
    archive or a preview, and nothing else. Any other uncommented line is a
    human's; the new paths are printed for them and nothing is touched. With
    a preview, the preview is the line that rides and the archive is a
    commented line beneath it; without one, the archive rides alone,
    labelled UNSANITIZED. THE FIRST ROUTER CARRIES THE VIM LESSON
    (2026-09-20): `first` is None to decide by absence, cached per process in
    _ROUTER_FIRST, or True/False to force it, as the probe does.
    """
[[[DIVIDER]]]
    THE ROUTER IS THE NEWCOMER'S FILE (2026-09-16): `context` opens it, so it is
    replaced ONLY while it is a file this writer could have produced -- the
    introduction line at most, plus one uncommented absolute path to an
    archive or a preview, and nothing else. A HUMAN'S FILE IS APPENDED TO
    (2026-09-25, operator's ruling): any other uncommented line means the
    file is theirs, so this walk lands as one marked block at the bottom,
    the lines above untouched, and a second write in the same ride replaces
    only its own block. With a preview, the preview is the line that rides
    and the archive is a commented line beneath it; without one, the archive
    rides alone, labelled UNSANITIZED. THE FIRST ROUTER CARRIES THE VIM

```
    LESSON (2026-09-20): `first` is None to decide by absence, cached per
    process in _ROUTER_FIRST, or True/False to force it, as the probe does.
    Returns (target, receipt): the path written and one sentence saying how.
    """
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
    # REPLACE ONLY A FILE THIS WRITER COULD HAVE WRITTEN: the introduction line
    # at most, plus one uncommented absolute path to an archive or a preview,
    # and nothing else. Any other uncommented line is a human's, so the paths
    # are printed for them instead.
    if target.is_file():
        owned = [line.strip() for line in target.read_text(encoding="utf-8").splitlines()
                 if line.strip() and not line.lstrip().startswith("#")]
        routed = [line for line in owned if line != str(INTRO_PATH)]
        if owned and (len(routed) != 1 or not routed[0].startswith("/")
                      or not routed[0].endswith(("captures.md", "decant-preview.md"))):
            raise ValueError(
                f"{target.name} has lines of your own, so nothing was written; add the line(s) yourself:
"
                + " ".join(line for line in (preview, source) if line)
                + " (or delete the file and the next walk rewrites it)")
[[[DIVIDER]]]
    # REPLACE ONLY A FILE THIS WRITER COULD HAVE WRITTEN: the introduction line
    # at most, plus one uncommented absolute path to an archive or a preview,
    # and nothing else. Any other file with lines in it is a human's, and a
    # human's file is APPENDED TO, never replaced (2026-09-25): one marked
    # block at the bottom, this walk's own, replaced in place when the same
    # ride writes twice, so their lines above never move.
    existing = ""
    appended = False
    if target.is_file():
        existing = target.read_text(encoding="utf-8")
        owned = [line.strip() for line in existing.splitlines()
                 if line.strip() and not line.lstrip().startswith("#")]
        routed = [line for line in owned if line != str(INTRO_PATH)]
        appended = not (owned and len(routed) == 1 and routed[0].startswith("/")
                        and routed[0].endswith(("captures.md", "decant-preview.md")))
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
    body = "\n".join(legend + ["", "# --- BEGIN LISTING FILES OR `! ` COMMANDS ---"] + listing)
+ "\n"
    body += ROUTER_TAIL if first else "\n# The next walk rewrites this file unless you have
edited it.\n"
    temp = None
[[[DIVIDER]]]
    body = "\n".join(legend + ["", "# --- BEGIN LISTING FILES OR `! ` COMMANDS ---"] + listing)
```

```
+ "\n"
    body += ROUTER_TAIL if first else "\n# Walks rewrite this file until you edit it; after that they
append below.\n"
    if appended:
        run_id = Path(archive_path).parent.name
        start, end = f"# --- WALK {run_id} START ---", f"# --- WALK {run_id} END ---"
        # The introduction describes a newcomer; the person whose lines sit
        # above is not one, so it rides commented here, one keystroke away.
        block_lines = [start] + legend + ["# --- BEGIN LISTING FILES OR `! ` COMMANDS ---"]
        block_lines += [f"# {line}" if line == intro else line for line in listing]
        block = "\n".join(block_lines + [end]) + "\n"
        at, to = existing.find(start), existing.find(end)
        if at != -1 and to > at:
            text = existing[:at] + block + existing[to + len(end):].lstrip("\n")
            receipt = "replaced this walk's own block at the bottom; your lines above are untouched"
        else:
            text = (existing.rstrip("\n") + "\n\n" if existing.strip() else "") + block
            receipt = f"appended WALK {run_id} at the bottom; your lines above are untouched"
    else:
        text = head + body
        receipt = ("0600; lists the summary; the archive is a commented line beneath it" if preview
                   else "0600; lists the UNSANITIZED archive until a summary is saved")
    temp = None
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
            prefix=".context-", delete=False,
        ) as stream:
            temp = Path(stream.name)
            os.fchmod(stream.fileno(), 0o600)
            stream.write(head + body)
            stream.flush()
            os.fsync(stream.fileno())
        os.replace(temp, target)
        temp = None
    finally:
        if temp is not None:
            temp.unlink(missing_ok=True)
    return target
[[[DIVIDER]]]
            prefix=".context-", delete=False,
        ) as stream:
            temp = Path(stream.name)
            os.fchmod(stream.fileno(), 0o600)
            stream.write(text)
            stream.flush()
            os.fsync(stream.fileno())
        os.replace(temp, target)
```

```
        temp = None
    finally:
        if temp is not None:
            temp.unlink(missing_ok=True)
    return target, receipt
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
        try:
            target = _write_walk_router(archive["path"])
        except Exception as exc:
            print(f"  CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
            print("  Archive preserved; the context file on disk is unchanged.")
        else:
            print(f"  CONTEXT FILE  {target}  (0600; lists the UNSANITIZED archive until a
summary is saved)")
[[[DIVIDER]]]
        try:
            target, receipt = _write_walk_router(archive["path"])
        except Exception as exc:
            print(f"  CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
            print("  Archive preserved; the context file on disk is unchanged.")
        else:
            print(f"  CONTEXT FILE  {target}  ({receipt})")
[[[REPLACE]]]
Target: scripts/mother_cat.py
[[[SEARCH]]]
            try:
                router = _write_walk_router(archive_path, preview_path=target)
            except Exception as exc:
                print(f"   CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
            else:
                print(f"   CONTEXT FILE  {router}  (0600; lists the summary; the archive is a
commented line beneath it)")
[[[DIVIDER]]]
            try:
                router, receipt = _write_walk_router(archive_path, preview_path=target)
            except Exception as exc:
                print(f"   CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
            else:
                print(f"   CONTEXT FILE  {router}  ({receipt})")
[[[REPLACE]]]
```

Ignition: type exit, then ndq, before the next compile (the flake sources .env on entry, and Car 2 changed it; the walk's writer and the sound loader load at call time).

4. PROMPT

THE AFTER READING for the rename-and-append ride, one probe at a time:

1. The adhoc census is a CENSUS this turn: foo_files.py 6, scripts/git_hooks/pre-commit 1, .gitignore 2, the connectors, GLOSSARY.md 3, the two SKILL.md files, two_arm.py 1 and ascii_displays.py 7, all unchanged.
2. The export: exactly one .env line, now ending in context.txt, and ENV= reading /home/mike/.local/state/pipulate/context.txt; the Processing Log's ROUTER LOADED line names context.txt.
3. The state directory lists context.txt and neither retired name.
4. SOUNDS None None before; two paths under assets/sounds after, with HOME pointed away so the home copies cannot answer for the repo.
5. git ls-files assets/sounds: 0 -> 3 (two wavs and the README).
6. The append fixture: the ValueError line before; APPEND_PROBE blocks=1 mine=1 lines=10 after, meaning two calls left one block, your line survived, and the file is your two lines, a blank, and a seven-line block. A real walk prints "appended WALK walk-... at the bottom" on its first write and "replaced this walk's own block" on its second; ride one whenever you like, and the transcript is the witness.
7. lines-of-your-own census: scripts/mother_cat.py 1 -> 0, foo_files.py 1 both taps; its HANDOFF paragraph says IT REPLACES, IT NEVER MERGES, and the next turn's car adds the append clause, with the file in context.
8. rg -nw cpr prompt_foo.py: two lines, both taps; the 2026-09-18 conviction stays as history, and the other gets a ruling.
Car receipts: HEADER_PRUNED with template=40 and tail_muted=12 (three files and nine probe echoes), then RENAMED adhoc.txt->context.txt env_lines_before=2 env_lines_after=1 adhocwalk_removed=True.

NEXT TURN'S CARS: the HANDOFF paragraph and the Selection line in foo_files.py chapter VIII-b (append, never replace, for a human's file); the bunny's new words registered in imports/ascii_displays.py as a three-region edit (render function, CRC ledger, registry) with figurate white_rabbit reading Drift: 0 as the ignition; a ruling on the second cpr line in prompt_foo.py.
RULINGS CARRIED: whether the reading frame gets a word back or stays compile --frame lean; whether PIPULATE_ADHOC_FILE, ADHOC_CHOP, the ADHOC SLOT markers and walk.py's adhoc_mutation key are renamed, each a gated car with its readers in context; whether page 3 of the walk names prompt and compile beside its question; the six dated adhoc lines in foo_files.py, for a forget ride; the GLOSSARY.md purge, a ride of its own.

5. EXTERNAL DELIVERABLES

None this turn outside the repo.
MikeLev.in: Wow, this is an epic response for a reason I'll explain when we get to that part. Oh and before we even get started it's worth pointing out this:
(nix) pipulate $ g

Before conducting experiment, type "blast" to establish left-hand causal boundary.
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified: foo_files.py

no changes added to commit (use "git add" and/or "git commit -a")
(nix) pipulate $ blast
📝 Committing: chore(router): set AI-edit blast boundary (foo_files.py context curation)
[main 1a740109] chore(router): set AI-edit blast boundary (foo_files.py context curation)
 1 file changed, 2 insertions(+), 2 deletions(-)
🚀 Pushing 1 commit(s) to remote...
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 48 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 390 bytes | 390.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:pipulate/pipulate.git
   8b4072b6..1a740109 main -> main

[A bunch of "blast radius" spaces deleted]

Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $

The system recognized on a g that I needed to blast and when I did the message adjusted accordingly. Excellent!

Establishing Causal Boundaries and Audit Invariants

Same commands, run twice, one change between them. Where the readings differ is what the change did; the diff in the middle is the receipt.
1: Probe:
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

```
(nix) pipulate $ rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
ls -1 ~/.local/state/pipulate/
HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file; print("SOUNDS",
_sound_file("tick.wav"), _sound_file("ding.wav"))' 2>&1 | tail -n 1
git ls-files assets/sounds | wc -l
PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib;
sys.path.insert(0,"scripts"); import mother_cat as m; p=pathlib.Path("/tmp/ctxprobe.txt");
p.write_text("# mine\nflake.nix\n"); a=pathlib.Path("/tmp/walk-probe/captures.md");
r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d
mine=%d lines=%d" % (t.count("WALK walk-probe START"), t.count("flake.nix"),
len(t.splitlines())))' 2>&1 | tail -n 1
for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER
MERGES' "$f" || echo 0)"; done
rg -nw cpr prompt_foo.py
./connectors/botify.py:1
./connectors/confluence.py:1
./connectors/gsc.py:1
./connectors/jira.py:1
./connectors/mcp.py:1
./connectors/README.md:1
./connectors/sheets.py:1
./connectors/slack.py:1
./connectors/wallet.py:3
./foo_files.py:6
./.gitignore:2
./GLOSSARY.md:3
./imports/ascii_displays.py:7
./Notebooks/.agents/skills/gsc_readonly/SKILL.md:1
./Notebooks/.agents/skills/sheets_readonly/SKILL.md:1
./scripts/git_hooks/pre-commit:1
./scripts/two_arm.py:1
/home/mike/repos/pipulate/.env
12:PIPULATE_ADHOC_FILE="$HOME/.local/state/pipulate/adhoc.txt"
13:PIPULATE_ADHOC_FILE="$HOME/.local/state/pipulate/adhoc.txt"
ENV=/home/mike/.local/state/pipulate/adhoc.txt
adhoc.txt
adhocwalk.txt
dedupe
facets.db
plan.json
SOUNDS None None
0
ValueError: ctxprobe.txt has lines of your own, so nothing was written; add the line(s) yourself:
/tmp/walk-probe/captures.md (or delete the file and the next walk rewrites it)
scripts/mother_cat.py:1
```

foo_files.py:0
1791:   CONVICTED 2026-09-18: a page question compiled through cpr inherited
2394:   # cpr included, rewrote this TRACKED file to read zero articles; the
(nix) pipulate $

2: Context:
The Not-Managed-by-Git Safe-for-Client-Data place. Insert Simpson Couch Gag in white space below (explain anything to the audience you feel needs it explained)
____   _   _
/ ___|___ _ __ | |_ _____ _| |_   Insert Simpson Couch Joke here.
| | / _ \| '_ \| __/ _ \ \/ / __| It's interesting shifting the identity here from adhoc.txt to context.txt.
| |__| (_) | | | | || __/> <| |_   Curiouser and curiouser!
_______/|_| |_|_____/_/___|
context.txt

To Human: WELCOME TO VIM! It's really NeoVim but I say vim anyway.
To AI: This is an alternate 40K view to the big book-ore rolling pin spine.

1. **Probe**: Baseline Reading
2. **Context**: Post-experiment *planned* reading instructions
3. **Patch**: The experiment and how to make it happen
4. **Prompt**: Post-experiment instructions and how to read results
5. **Deliverable**: How the world is forever different moving forward

The first thing you need to know here is that everything that comes after the
hash symbol (#) is commented out — and that's EVERYTHING in this file's default
state. Begin editing-in lines for inclusion as part of the context or adding
chunks of new context at the bottom. `Ctrl`+`v`, `j` (repeatedly), `l` (to move
right), `d` (to delete). Reverse that with `Ctrl`+`v`, `j` (repeatedly),
`Shift`+`i`, `# `, `Esc` to put the hashes back. You can just arrow-key around
here with `h`, `j`, `k`, `l`. Save-and-quit is a bit tricky because another
file is also loaded: `Esc`, `:`, `q`, `w`, `!`

If this is stressing you out and you're a quitter and want to quit, just type:
`Esc`, `:`, `q`, `!`, `Enter`. That will exit without saving any changes. If
you want to get over this hump, type: `Esc`, `:`, `T`, `u`, `t`, `o`, `r`, `Enter`.

This file is just to make it easy having options of what to edit into context.
You can use whatever text-file you want to stack file-names and commands to
build an output text-file with the identically stacked output of each file or
command. In this way we vertically append or "stack" a bunch of text; simple as
that. If you understand this concept, you're on your way to future-proofing
yourself in the Age of AI. Congratulations! Here is how to include web pages:

!URL --
when   Public page; what a stranger or crawler sees; the BEFORE of a
login-wall diagnosis

```
#     switch  It shows a login page -> `warm URL` once, then `?URL`
#
#   ?URL  ------------------------------------------------------------------
#     when    Anything behind a login, on the site's persistent profile;
#             `check URL` first
#     switch  The lenses show a shell (nav, an `[Iframe]` leaf, no content) ->
#             read the wire truth for the XHR the frame makes, then call that
#             API with a connector
#
#   @URL  ------------------------------------------------------------------
#     when    Every re-read of a page already scraped; no browser, no network
#     switch  The cached page is stale or was a login wall -> fresh `!` or `?`
#
#   $URL  ------------------------------------------------------------------
#     when    Exact markup: meta tags, a JSON blob in a `<script>`
#     note    Token-heavy; needs a prior scrape
#
#   %URL  ------------------------------------------------------------------
#     when    The network log distilled; SPA endpoint discovery
#     switch  It re-serves the wire truth you already have -> the API
#
#   ! cmd  -----------------------------------------------------------------
#     when    Any bounded, non-interactive command as a live receipt
#     note    Cap it with `-n`; no aliases, no prompts
#
#   Connector  -------------------------------------------------------------
#     when    The number you want is one GET away
#     switch  LIST until the thing isn't in the list -> FETCH by id -> DRILL
#             the path the app's own frame called -> `--grep` to narrow a list
#             or find a leaf

# Every step is one argument longer than the last; the moment a lens shows less than the wire does is the moment to stop scraping.

# FOR 40K-FT VIEW (STORY & INFRASTRUCTURE) !!
# --- START EDITING-IN ON 1ST TURN ---

# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- ROLLING PIN that gives the 40K foot book-spine view of book-ore (only works for me because of local-only git repo)
# ~/repos/nixos/autognome.py  # <-- You wake up in the morning and your Tooling & Instrumentation folds out of you like Inspector Gadget.
# init.lua                    # <-- Those gadgets are made easy-to-use through nifty keyboard shortcuts (but ya gotta learn vim).
# assets/installer/install.sh # <-- Pipulate.com installer real home in github/pipulate repo
# # GLOSSARY.md               # <-- Like the back of a J.R.R. Tolkien book, there's kooky new terms to know.
# flake.nix                   # <-- Here is my hardware. Here is my state. Put on your sandbox. And please recreate. (Infrastructure as Code / IaC)
```

```
# prompt_foo.py              # <-- THIS system
# foo_files.py               # <-- main ROUTER
# requirements.in            # <-- We've "pinned" everything but still want a flexible Python Data Science virtualenv.
# pyproject.toml             # <-- How this is a citizen of the Python "pip install" ecosystem
# __init__.py                # <-- Version info

# --- END EDITING-IN ON 1ST TURN ---

# scripts/articles/lsa.py     # <-- 2ND BRAIN: Search external memory with `rgx`, `rgxc` & `posts` Blogging for Hackers Jekyll-compatible.

# OPTIONAL ACTUATORS (cheap and good to include to expand the AI's capabilities)
# cli.py                     # <-- Catch-all actuator for PyPI envs, Python anchoring, MCP tool-call (plus alternatives) and **kwargs like wrapping for CLI
# scripts/xp.py              # <-- Transforms host OS copy-paste buffer player-piano music into context-payload.
# scripts/ai.py              # <-- How I constantly use local AI to write git commit messages with `m` alias.
# scripts/crawl.py           # <-- Feel free to ask for something to be crawled and included in the next turn.
# scripts/weblogin.py        # <-- Lets the user "warm up" the cache for their web logins at their leisure on a profile that persists.
# scripts/webclip_2_markdown.py  # <-- Surprisingly important program.

# MISCELLANEOUS (rare to include but sometimes critical)
# scripts/foo_cartridge.py    # Needs description
# scripts/foo_replay.py       # Needs description
# release.py                 # <-- How everything ends up where it does (GitHub, PyPI, etc.)
# imports/voice_synthesis.py  # <-- The wand can talk to you
# imports/ascii_displays.py   # <-- Where all the ASCII Art lives
# scripts/release/version_sync.py  # <-- Needs to be wrapped into release.py and eliminated, I think.

#                     --- Under this line is were you paste what the AI gives you ---
#                     --- We call it context but it's really just the right-hand  ---
#                     --- blast-radius of the "probes" to make this all science.  ---

# Carry-over as the important work-in-progress parts of the project here just
# like above but not as long-standing overarching to the framework but rather
# for the current hot spots actively being worked on.

# --- START THIS DISCUSSION ---

# Get things started here! Guess at what context should be included.
# If you get it wrong, you're just wasting 1-turn because the AI will help.
# Un-comment lines, add lines with absolute-path filenames or `! ` commands.
```

Context 1 (Edit-in selections from above and add new files immediately below)
LAUNCH
walk   # <-- Repo-root wrapper; delegates to mck.sh, NOT to scripts/walk.py
assets/installer/mck.sh   # <-- THE LAUNCHER (curl -fsSL pipulate.com/mck.sh | bash): finds the workshop, forces the spoken rehearsal on first contact, asks for the word RIDE
scripts/mother_cat.py   # <-- THE RIDER (alias: mothercat), Car B: actuates walk.py's validated plan; per-run capture banking, DECANT, the adhocwalk.txt writer
scripts/boot_menu.py   # <-- The one-door command list; short on shell entry, expanded by `all`, both rendered from tuples in the file
scripts/sources_menu.py   # <-- The roster `conn` prints; filename stays descriptive, typed word is shorter
#
COMPILE
scripts/walk_compile.py   # <-- .walk.md -> <name>.yaml BESIDE the surface, and only where git ignores it; url_env stops only (it refuses a scheme separator anywhere in its output), so it is the PRIVATE lane's compiler and a bundled trail in assets/trails/ is hand-written JSON; refuses on every TODO left in the surface; stdlib only, imports nothing
#
VALIDATE
scripts/walk.py   # <-- Car A, the planner: loads a trail, refuses unknown or missing keys in BOTH directions, prints the dry-run plan; holds DEFAULT_TRAIL; never actuates
#
SEAL
scripts/walk_cartridge.py   # <-- trail -> data/walks/<sha256>/walk.zip (trail + manifest with hash and consent surface); reseal is byte-identical
#
CAPTURE
scripts/weblogin.py   # <-- SETTLE ahead of time: warm a login on the persistent profile; --profile default unless told otherwise
tools/scraper_tools.py   # <-- The browser capture and the CAPTURE fence (_capture_checkpoint is a write barrier, not a prompt)
imports/voice_synthesis.py # <-- Piper: scripted narration, never a model (ATTRIBUTED-VOICE)
#
HAND OFF
prompt_foo.py   # <-- Compiles the selected router (adhocwalk.txt: `epr` edits it, `cpr` compiles it) into a payload
scripts/foo_cartridge.py   # <-- The second seal, over the EVIDENCE: payload.md, prompt.md, manifest.json
scripts/foo_replay.py   # <-- Context extraction and attention checks; not browser or API replay
tests/test_mck_rep2.py   # <-- Rep 2: the earmark's owed side-by-side witness
#
THE TRAILS (bundled; each status is the newest receipt, never a promise)
assets/trails/public_walk.json   # profile default; SETTLE trivial; three unlinked npvg.org pages (the_word, the_receipt, the_two_pages), inline script on stop three only, connector noop.py; RIDDEN 2026-09-15 on these stops, 3 of 3 at artifacts=11 each, archive complete, DECANT AUTHORIZED at 20,992 B with checks 0/0/0, and the chatbot given stop three's

question said the preview does not carry the server's sentence, which is true (the diff-lens TODO); the 2026-08-01 ride walked the OLD stop set (example.com, mikelev.in, pipulate.com); what bare `walk` rides
assets/trails/practice.yaml   # one configurable page; UNREAD, label stale since 2026-08-01
assets/trails/first_context.yaml   # profile default; SETTLE real; Jira/Botify/Gmail, THREE wallet kinds in one trail; UNRIDDEN
assets/trails/botify_pageworkers.yaml # profile botify; SETTLE real; UNRIDDEN -- the only trail on a non-default profile, and no such ride has ever been witnessed
assets/trails/jira_for_you.yaml   # profile default; SETTLE real (Atlassian sign-in); top of the ticket loop; RIDDEN 2026-09-01 and DECANTED, the sign-in moment unrecorded (a cold profile records it)
assets/trails/ticket.yaml   # profile default; SETTLE real; url_env x2 (JIRA, BOTIFY) fed by one gitignored exports file per ticket under Notebooks/Client_Work/tickets/; stop two of the ticket loop; RIDDEN 2026-09-01 to stop 1 then REFUSED at stop 2 on an unset url_env, Linux and Mac -- the ride that bought the rider its PRE-FLIGHT. RULING: the connector does NOT run at DECANT (DECANT gates only CAPTURE-fenced material; API auth is a different wallet; the rider never harvests). The bridge to the issue text is a ! line in adhoc.txt.
assets/trails/se_ticket.yaml   # profile default; SETTLE real; the SE ticket template: for_you (literal) + issue (JIRA, required) + project/slack/confluence (optional, skipped when unset); RIDDEN 2026-09-02, 3 of 5 captured, 2 skipped, 72,383 B decanted, card showed both rows; harvest regexes on optional stops are PLACEHOLDERS (botify_analysis's already fullmatched a live analysisSlug); guidance strings hand-count "of five" and go stale on the sixth stop
#
UNLINKED PAGES (npvg.org; nothing links to them and no page links out, so a trail is the only edge; nixops.sh rsyncs them, no rebuild)
remotes/honeybot/www/npvg.org/index.html   # <-- the door: a browser gets this page, curl and wget get install.sh ($npvg_index)
remotes/honeybot/www/npvg.org/walk/1/index.html   # <-- public_walk stop one: the capture word; no script, so source and hydrated DOM should match
remotes/honeybot/www/npvg.org/walk/2/index.html   # <-- stop two: count the archive's fingerprints against the terminal's artifacts= number
remotes/honeybot/www/npvg.org/walk/3/index.html   # <-- stop three: the walk's only script rewrites the server's sentence and appends a paragraph; the DECANT test lives here
#
OFF-ROSTER DISTRIBUTION RESIDUE (not a walk dependency)
assets/installer/replay.sh # <-- OFF the roster 2026-08-01, stranded; re-add needs syntax + one ride + a pinned verifier fetch

Context 2
--- THIS RIDE: the files the cars touched, so the AFTER can read them ---
.gitignore
prompt_foo.py
apply.py
scripts/mother_cat.py
scripts/boot_menu.py
init.lua

```
# flake.nix
# foo_files.py
# # --- NEXT-TURN CARS: the rename's remaining homes; the census names which ---
# scripts/xp.py
# scripts/git_hooks/pre-commit
# introduction.md
# scripts/walk.py
# scripts/sources_menu.py
# # GLOSSARY.md
# # README.md
# # AGENTS.md
# # --- THE 40K-FOOT VIEW (your call) ---
# # ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs
# # --- PROBE ECHOES ---
# ! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
# ! rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
# ! .venv/bin/python scripts/boot_menu.py --recall
# ! rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type
context' scripts/mother_cat.py
# ! rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
# ! awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END
.(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
# ! git check-ignore -v context.txt || echo NOT_IGNORED
# ! for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f
"$f" && rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
# ! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2

# Context 3
# --- NEXT TURN'S CARS: the rename (once line 2 names the export), the sounds, the walk's
write ---
# scripts/mother_cat.py
# tools/scraper_tools.py
# .gitignore
# # flake.nix        # <-- only if Car 8 or 9 refused; line 7 is their witness
# # foo_files.py     # <-- only if Car 10 refused, or to rule on the six dated adhoc lines it keeps
# # init.lua         # <-- only if Car 6 refused
# # introduction.md  # <-- only if Car 3 refused
# # --- PROBE ECHOES ---
# ! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
# ! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
# ! for f in scripts/git_hooks/pre-commit .git/hooks/pre-commit; do echo "$f:$(rg -c 'context\.txt'
"$f" || echo 0)"; done
# ! rg -cw 'epr|cpr' --sort path prompt_foo.py foo_files.py flake.nix init.lua introduction.md
GLOSSARY.md scripts assets/installer remotes/honeybot/www/npvg.org
# ! for f in scripts/walk.py init.lua scripts/mother_cat.py; do echo "$f:$(rg -c adhoc "$f" || echo 0)";
```

```
done
# ! rg -n 'door 2' scripts/sources_menu.py || echo door2=0
# ! LD_LIBRARY_PATH="" nix eval --raw .#devShells.x86_64-linux.quiet.shellHook 2>/dev/null |
rg -c 'Before conducting experiment' || echo 0
# ! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
# ! ls -la ~/.local/share/pipulate/; rg -n 'local/share/pipulate|tick\.wav|ding\.wav|\*\.wav'
tools/scraper_tools.py .gitignore

# Context 4
# --- NEXT TURN'S CARS: the HANDOFF paragraph, the bunny, the cpr survivors ---
foo_files.py
imports/ascii_displays.py
prompt_foo.py
# scripts/mother_cat.py    # <-- only if Car 5 refused; line 6 is its witness
# tools/scraper_tools.py   # <-- only if Car 3 refused; line 4 is its witness
# .gitignore
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
! ls -1 ~/.local/state/pipulate/
! HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file; print("SOUNDS",
_sound_file("tick.wav"), _sound_file("ding.wav"))' 2>&1 | tail -n 1
! git ls-files assets/sounds | wc -l
! PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib;
sys.path.insert(0,"scripts"); import mother_cat as m; p=pathlib.Path("/tmp/ctxprobe.txt");
p.write_text("# mine\nflake.nix\n"); a=pathlib.Path("/tmp/walk-probe/captures.md");
r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d
mine=%d lines=%d" % (t.count("WALK walk-probe START"), t.count("flake.nix"),
len(t.splitlines())))' 2>&1 | tail -n 1
! for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER
MERGES' "$f" || echo 0)"; done
! rg -nw cpr prompt_foo.py
```

3: Patches:

```
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ python - <<'EOF'
import os, sys
from pathlib import Path
root = Path(os.environ.get("PIPULATE_ROOT", "."))
sel = os.environ.get("PIPULATE_ADHOC_FILE", "")
p = Path(sel).expanduser() if sel.strip() else root / "context.txt"
if not p.is_file():
```

```
    sys.exit(f"STOP: no context file at {p}")
ff = (root / "foo_files.py").read_text(encoding="utf-8").split("\n")
S = [i for i, l in enumerate(ff) if l.rstrip() == "# --- START `context.txt` TEMPLATE ---"]
E = [i for i, l in enumerate(ff) if l.rstrip() == "# --- END `context.txt` TEMPLATE ---"]
assert len(S) == 1 and len(E) == 1 and S[0] < E[0], ("STOP: template markers", S, E)
template = ff[S[0] + 1:E[0]]
while template and not template[-1].strip():
    template.pop()
lines = p.read_text(encoding="utf-8").split("\n")
if lines[:1] == template[:1]:
    print(f"HEADER_ALREADY_PRUNED {p}; nothing written")
    sys.exit(0)
C = [i for i, l in enumerate(lines) if l.startswith("# Context 1 (Edit-in")]
assert len(C) == 1, ("STOP: Context 1 marker", C)
tail, muted = [], 0
for l in lines[C[0] + 1:]:
    if l.strip() and not l.lstrip().startswith("#"):
        l, muted = "# " + l, muted + 1
    tail.append(l)
out = template + ["", "# Context 1"] + tail
p.write_text("\n".join(out), encoding="utf-8")
print(f"HEADER_PRUNED {p} header_lines_replaced={C[0] + 1} template={len(template)}
tail_kept={len(tail)} tail_muted={muted}")
EOF
HEADER_PRUNED /home/mike/.local/state/pipulate/adhoc.txt header_lines_replaced=123
template=40 tail_kept=115 tail_muted=11
(nix) pipulate $ context

Well that changes context! Wow, okay. I like this.
# context.txt: the list of files an AI will read. context opens it, compile builds it.
# Keys: j down, k up; Esc then :q quits, :q! discards, :wq saves and quits.
# One line per thing the AI reads: a file path, or a command after `! `.
# A line that starts with # is a comment: that file is not read.
# To add a line: i starts typing, Esc stops. Absolute paths work from anywhere.
# Web pages, APIs and the connector words: chapter XVIII of foo_files.py.

# --- THE 40K-FOOT VIEW (uncomment on a first turn; comment out on the second) ---
# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- the book's spine, one line
per article, newest first
# ~/repos/nixos/autognome.py  # <-- the machine's morning routine (this author's NixOS box
only)
# init.lua                  # <-- the editor keys that drive the day
# assets/installer/install.sh # <-- how a stranger's machine gets this workshop
# GLOSSARY.md                 # <-- the terms, defined
# flake.nix                 # <-- the environment, pinned: here is my hardware, here is my state
# prompt_foo.py               # <-- the compiler that builds the payload
# foo_files.py              # <-- the router: which files ride, and this book's outline
# scripts/articles/lsa.py     # <-- the second brain: the article corpus behind `rgx`, `rgxc` and
```

`posts`
requirements.in   # <-- the Python packages, pinned
pyproject.toml   # <-- the PyPI package
__init__.py   # <-- the version

--- ACTUATORS (cheap; include when the AI should be able to act, not only read) ---
cli.py   # <-- tool calls from the command line
scripts/xp.py   # <-- turns a pasted reply into the next context
scripts/ai.py   # <-- a local AI writes the commit messages
scripts/crawl.py   # <-- crawl a site into the next turn
scripts/weblogin.py   # <-- warm a login on the persistent browser profile
scripts/webclip_2_markdown.py # <-- a web page, clipped, as markdown

--- RARE ---
scripts/foo_cartridge.py   # <-- the sealed archive: writer and verifier
scripts/foo_replay.py   # <-- replay a sealed archive on another machine
release.py   # <-- how a release reaches GitHub and PyPI
imports/voice_synthesis.py # <-- the voice
imports/ascii_displays.py   # <-- the ASCII art
scripts/release/version_sync.py # <-- version stamping (to be folded into release.py)

--- THIS DISCUSSION ---
The files and commands for the work in front of you. Paste the NEXT CONTEXT
block an AI hands back directly below this line; the AI will correct a guess.

Context 1
LAUNCH
walk   # <-- Repo-root wrapper; delegates to mck.sh, NOT to scripts/walk.py
assets/installer/mck.sh   # <-- THE LAUNCHER (curl -fsSL pipulate.com/mck.sh | bash): finds the workshop, forces the spoken rehearsal on first contact, asks for the word RIDE
scripts/mother_cat.py   # <-- THE RIDER (alias: mothercat), Car B: actuates walk.py's validated plan; per-run capture banking, DECANT, the adhocwalk.txt writer
scripts/boot_menu.py   # <-- The one-door command list; short on shell entry, expanded by `all`, both rendered from tuples in the file
scripts/sources_menu.py   # <-- The roster `conn` prints; filename stays descriptive, typed word is shorter
#
COMPILE
scripts/walk_compile.py   # <-- .walk.md -> <name>.yaml BESIDE the surface, and only where git ignores it; url_env stops only (it refuses a scheme separator anywhere in its output), so it is the PRIVATE lane's compiler and a bundled trail in assets/trails/ is hand-written JSON; refuses on every TODO left in the surface; stdlib only, imports nothing
#
VALIDATE
scripts/walk.py   # <-- Car A, the planner: loads a trail, refuses unknown or missing keys in BOTH directions, prints the dry-run plan; holds DEFAULT_TRAIL; never actuates
#
SEAL

scripts/walk_cartridge.py   # <-- trail -> data/walks/<sha256>/walk.zip (trail + manifest with hash and consent surface); reseal is byte-identical
#
CAPTURE
scripts/weblogin.py   # <-- SETTLE ahead of time: warm a login on the persistent profile; --profile default unless told otherwise
tools/scraper_tools.py   # <-- The browser capture and the CAPTURE fence (_capture_checkpoint is a write barrier, not a prompt)
imports/voice_synthesis.py # <-- Piper: scripted narration, never a model (ATTRIBUTED-VOICE)
#
HAND OFF
prompt_foo.py   # <-- Compiles the selected router (adhocwalk.txt: `epr` edits it, `cpr` compiles it) into a payload
scripts/foo_cartridge.py   # <-- The second seal, over the EVIDENCE: payload.md, prompt.md, manifest.json
scripts/foo_replay.py   # <-- Context extraction and attention checks; not browser or API replay
tests/test_mck_rep2.py   # <-- Rep 2: the earmark's owed side-by-side witness
#
THE TRAILS (bundled; each status is the newest receipt, never a promise)
assets/trails/public_walk.json   # profile default; SETTLE trivial; three unlinked npvg.org pages (the_word, the_receipt, the_two_pages), inline script on stop three only, connector noop.py; RIDDEN 2026-09-15 on these stops, 3 of 3 at artifacts=11 each, archive complete, DECANT AUTHORIZED at 20,992 B with checks 0/0/0, and the chatbot given stop three's question said the preview does not carry the server's sentence, which is true (the diff-lens TODO); the 2026-08-01 ride walked the OLD stop set (example.com, mikelev.in, pipulate.com); what bare `walk` rides
assets/trails/practice.yaml   # one configurable page; UNREAD, label stale since 2026-08-01
assets/trails/first_context.yaml   # profile default; SETTLE real; Jira/Botify/Gmail, THREE wallet kinds in one trail; UNRIDDEN
assets/trails/botify_pageworkers.yaml # profile botify; SETTLE real; UNRIDDEN -- the only trail on a non-default profile, and no such ride has ever been witnessed
assets/trails/jira_for_you.yaml   # profile default; SETTLE real (Atlassian sign-in); top of the ticket loop; RIDDEN 2026-09-01 and DECANTED, the sign-in moment unrecorded (a cold profile records it)
assets/trails/ticket.yaml   # profile default; SETTLE real; url_env x2 (JIRA, BOTIFY) fed by one gitignored exports file per ticket under Notebooks/Client_Work/tickets/; stop two of the ticket loop; RIDDEN 2026-09-01 to stop 1 then REFUSED at stop 2 on an unset url_env, Linux and Mac -- the ride that bought the rider its PRE-FLIGHT. RULING: the connector does NOT run at DECANT (DECANT gates only CAPTURE-fenced material; API auth is a different wallet; the rider never harvests). The bridge to the issue text is a ! line in adhoc.txt.
assets/trails/se_ticket.yaml   # profile default; SETTLE real; the SE ticket template: for_you (literal) + issue (JIRA, required) + project/slack/confluence (optional, skipped when unset); RIDDEN 2026-09-02, 3 of 5 captured, 2 skipped, 72,383 B decanted, card showed both rows; harvest regexes on optional stops are PLACEHOLDERS (botify_analysis's already fullmatched a live analysisSlug); guidance strings hand-count "of five" and go stale on the sixth

stop
#
UNLINKED PAGES (npvg.org; nothing links to them and no page links out, so a trail is the only edge; nixops.sh rsyncs them, no rebuild)
remotes/honeybot/www/npvg.org/index.html   # <-- the door: a browser gets this page, curl and wget get install.sh ($npvg_index)
remotes/honeybot/www/npvg.org/walk/1/index.html   # <-- public_walk stop one: the capture word; no script, so source and hydrated DOM should match
remotes/honeybot/www/npvg.org/walk/2/index.html   # <-- stop two: count the archive's fingerprints against the terminal's artifacts= number
remotes/honeybot/www/npvg.org/walk/3/index.html   # <-- stop three: the walk's only script rewrites the server's sentence and appends a paragraph; the DECANT test lives here
#
OFF-ROSTER DISTRIBUTION RESIDUE (not a walk dependency)
assets/installer/replay.sh # <-- OFF the roster 2026-08-01, stranded; re-add needs syntax + one ride + a pinned verifier fetch

Context 2
--- THIS RIDE: the files the cars touched, so the AFTER can read them ---
.gitignore
prompt_foo.py
apply.py
scripts/mother_cat.py
scripts/boot_menu.py
init.lua
flake.nix
foo_files.py
--- NEXT-TURN CARS: the rename's remaining homes; the census names which ---
scripts/xp.py
scripts/git_hooks/pre-commit
introduction.md
scripts/walk.py
scripts/sources_menu.py
GLOSSARY.md
README.md
AGENTS.md
--- THE 40K-FOOT VIEW (your call) ---
! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs
--- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
! .venv/bin/python scripts/boot_menu.py --recall
! rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type context' scripts/mother_cat.py
! rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
! awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END .(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
! git check-ignore -v context.txt || echo NOT_IGNORED

```
# ! for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f
"$f" && rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
# ! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2

# Context 3
# --- NEXT TURN'S CARS: the rename (once line 2 names the export), the sounds, the walk's
write ---
# scripts/mother_cat.py
# tools/scraper_tools.py
# .gitignore
# # flake.nix        # <-- only if Car 8 or 9 refused; line 7 is their witness
# # foo_files.py     # <-- only if Car 10 refused, or to rule on the six dated adhoc lines it keeps
# # init.lua         # <-- only if Car 6 refused
# # introduction.md  # <-- only if Car 3 refused
# # --- PROBE ECHOES ---
# ! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
# ! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
# ! for f in scripts/git_hooks/pre-commit .git/hooks/pre-commit; do echo "$f:$(rg -c 'context\.txt'
"$f" || echo 0)"; done
# ! rg -cw 'epr|cpr' --sort path prompt_foo.py foo_files.py flake.nix init.lua introduction.md
GLOSSARY.md scripts assets/installer remotes/honeybot/www/npvg.org
# ! for f in scripts/walk.py init.lua scripts/mother_cat.py; do echo "$f:$(rg -c adhoc "$f" || echo 0)";
done
# ! rg -n 'door 2' scripts/sources_menu.py || echo door2=0
# ! LD_LIBRARY_PATH="" nix eval --raw .#devShells.x86_64-linux.quiet.shellHook 2>/dev/null |
rg -c 'Before conducting experiment' || echo 0
# ! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
# ! ls -la ~/.local/share/pipulate/; rg -n 'local/share/pipulate|tick\.wav|ding\.wav|\*\.wav'
tools/scraper_tools.py .gitignore

# Context 4
# --- NEXT TURN'S CARS: the HANDOFF paragraph, the bunny, the cpr survivors ---
foo_files.py
imports/ascii_displays.py
prompt_foo.py
# scripts/mother_cat.py    # <-- only if Car 5 refused; line 6 is its witness
# tools/scraper_tools.py   # <-- only if Car 3 refused; line 4 is its witness
# .gitignore
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
! ls -1 ~/.local/state/pipulate/
! HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file; print("SOUNDS",
```

```
_sound_file("tick.wav"), _sound_file("ding.wav"))' 2>&1 | tail -n 1
! git ls-files assets/sounds | wc -l
! PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib;
sys.path.insert(0,"scripts"); import mother_cat as m; p=pathlib.Path("/tmp/ctxprobe.txt");
p.write_text("# mine\nflake.nix\n"); a=pathlib.Path("/tmp/walk-probe/captures.md");
r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d
mine=%d lines=%d" % (t.count("WALK walk-probe START"), t.count("flake.nix"),
len(t.splitlines())))' 2>&1 | tail -n 1
! for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER
MERGES' "$f" || echo 0)"; done
! rg -nw cpr prompt_foo.py
```

That's nice and clean. I commented back in Context 4 so it will be part of the next compiled context. And we continue.

```
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ python - <<'EOF'
import os, sys
from pathlib import Path
root = Path(os.environ.get("PIPULATE_ROOT", "."))
env = root / ".env"
old, new = "adhoc.txt", "context.txt"
state = Path.home() / ".local/state/pipulate"
src, dst, walk = state / old, state / new, state / "adhocwalk.txt"
lines = env.read_text(encoding="utf-8").split("\n")
hits = [i for i, l in enumerate(lines) if l.startswith("PIPULATE_ADHOC_FILE=")]
bad = [i for i in hits if old not in lines[i]]
assert hits and not bad, ("STOP: .env export lines", len(hits), len(bad))
assert src.is_file() and not dst.exists(), ("STOP: state files", src.is_file(), dst.exists())
keep = lines[hits[0]].replace(old, new)
out = [l for i, l in enumerate(lines) if i not in hits[1:]]
out[hits[0]] = keep
env.write_text("\n".join(out), encoding="utf-8")
src.rename(dst)
removed = walk.exists()
if removed:
    walk.unlink()
print(f"RENAMED {src.name}->{dst.name} env_lines_before={len(hits)} env_lines_after=1
adhocwalk_removed={removed}")
EOF
RENAMED adhoc.txt->context.txt env_lines_before=2 env_lines_after=1
adhocwalk_removed=True
(nix) pipulate $
```

Wow, there's the file rename I was expecting. Now an exact interlock patch:

```
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ patch
(nix) pipulate $ app
✅ WHOLE-FILE WRITE: CREATED 'assets/sounds/README.md'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated '.gitignore'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'tools/scraper_tools.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'tools/scraper_tools.py'.
(nix) pipulate $ d
diff --git a/.gitignore b/.gitignore
index 787741ae..71a09783 100644
--- a/.gitignore
+++ b/.gitignore
@@ -11,6 +11,8 @@ Notebooks/Advanced_Notebooks/Whatsa_Lambda.ipynb
 patch
 *_true_schema_discovery.json
 *.wav
+# the walk's two CC0 sounds ship in the repo; assets/sounds/README.md names their
sources
+!assets/sounds/*.wav
 *.tsv
 # end temp

diff --git a/tools/scraper_tools.py b/tools/scraper_tools.py
index f8da224d..b61a30fc 100644
--- a/tools/scraper_tools.py
+++ b/tools/scraper_tools.py
@@ -72,16 +72,26 @@ def _guided_path_component(url: str) -> tuple[str, str]:


 # --- Optional audio during browser startup ---
-# Operator-owned courtesy sounds live outside the repo. tick.wav loops while
-# the browser is opening; ding.wav plays once when the page-load phase ends.
-# No file means no sound. Linux uses aplay; Darwin uses afplay.
+# tick.wav loops while the browser is opening; ding.wav plays once when the
+# page-load phase ends. Both are CC0 and ship in assets/sounds/ (its README
+# names the sources) since 2026-09-26, after a week in which the Mac was
+# silent because the only copies lived under one home directory. A copy
+# under ~/.local/share/pipulate still wins, so an operator can swap in their
+# own. No file in either place means no sound. Linux uses aplay; Darwin
+# uses afplay.
 SCRAPE_MUSIC_MARKER = "pipulate-scrape-music"
-SOUND_DIR = Path.home() / ".local/share/pipulate"
+SOUND_DIRS = (
+    Path.home() / ".local/share/pipulate",
```

```
+    Path(__file__).resolve().parent.parent / "assets" / "sounds",
+)


 def _sound_file(name):
-    path = SOUND_DIR / name
-    return path if path.is_file() else None
+    for sound_dir in SOUND_DIRS:
+        path = sound_dir / name
+        if path.is_file():
+            return path
+    return None


 def _player_args(sound):

--- UNTRACKED (invisible to the diff above; m will stage these) ---
  + assets/sounds/README.md
(nix) pipulate $ m
📝 Committing: chore: Update sound paths and documentation
[main 2b41b0ea] chore: Update sound paths and documentation
 2 files changed, 18 insertions(+), 6 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 48 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 940 bytes | 940.00 KiB/s, done.
Total 5 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To github.com:pipulate/pipulate.git
   1a740109..2b41b0ea  main -> main
(nix) pipulate $
```

Okay now I see we make the clock ticking and timer ding sounds of the browser automation part of the repo.

```
Before conducting experiment, type "blast" to establish left-hand causal boundary.
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    assets/sounds/

nothing added to commit but untracked files present (use "git add" to track)
(nix) pipulate $ cp ~/.local/share/pipulate/tick.wav ~/.local/share/pipulate/ding.wav
assets/sounds/ && ls -l assets/sounds/
total 1072
```

```
-rw-r--r-- 1 mike users 444494 Sep 26 01:05 ding.wav
-rw-r--r-- 1 mike users    646 Sep 26 01:04 README.md
-rw-r--r-- 1 mike users 645230 Sep 26 01:05 tick.wav
(nix) pipulate $ d

--- UNTRACKED (invisible to the diff above; m will stage these) ---
  + assets/sounds/README.md
  + assets/sounds/ding.wav
  + assets/sounds/tick.wav
(nix) pipulate $ git add assets/sounds/*
(nix) pipulate $ d
(nix) pipulate $ m
📝 Committing: chore: Add sound files for guided walk and page open events
[main a23b072c] chore: Add sound files for guided walk and page open events
 3 files changed, 14 insertions(+)
 create mode 100644 assets/sounds/README.md
 create mode 100644 assets/sounds/ding.wav
 create mode 100644 assets/sounds/tick.wav
(nix) pipulate $ git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 48 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 335.15 KiB | 7.79 MiB/s, done.
Total 7 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:pipulate/pipulate.git
   2b41b0ea..a23b072c  main -> main
(nix) pipulate $
```

Remember Captain Hook hearing the alligator that swallowed the clock tick? This is sort of like that but while waiting for web pages in browsers to settle during automation.

```
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'scripts/mother_cat.py'.
```

```
(nix) pipulate $ d
diff --git a/scripts/mother_cat.py b/scripts/mother_cat.py
index d2463c82..d974a6db 100644
--- a/scripts/mother_cat.py
+++ b/scripts/mother_cat.py
@@ -368,15 +368,15 @@ ROUTER_TAIL = (
     "#   3. type  compile  to build this list and the question into one payload\n"
     "#   4. paste your clipboard into a chatbot; compile filled it\n"
     "# To add a line later: i starts typing, Esc stops, :wq saves and leaves.\n"
-    "# The next walk rewrites this file (shorter notes) unless you have edited it.\n"
+    "# Walks rewrite this file until you edit it; after that they append below.\n"
 )
 # THE WHEELS COME OFF BY THEMSELVES. "First" means the router did not exist
 # before this ride. The rider writes it twice per ride (after ARCHIVE STATUS,
 # then after DECANT), so the absence test runs once per process and its
 # answer is cached here; the second write must not find the file its own
 # first write made. A later walk finds the file, writes ROUTER_KEYS, and the
-# wheels are off. A human who edits the router keeps it (the guard in the
-# writer); one who deletes it gets the lesson back on the next walk.
+# wheels are off. A human who edits the router keeps it: later walks append
+# a marked block below their lines; deleting it brings the lesson back.
 _ROUTER_FIRST = {"first": None}


@@ -390,18 +390,21 @@ def _router_line(path, what):


 def _write_walk_router(archive_path, preview_path=None, first=None):
-    """Replace one local selection; never read, disclose or compile its evidence.
+    """Write one local selection; never read, disclose or compile its evidence.

     THE ROUTER IS THE NEWCOMER'S FILE (2026-09-16): `context` opens it, so it is
     replaced ONLY while it is a file this writer could have produced -- the
     introduction line at most, plus one uncommented absolute path to an
-    archive or a preview, and nothing else. Any other uncommented line is a
-    human's; the new paths are printed for them and nothing is touched. With
-    a preview, the preview is the line that rides and the archive is a
-    commented line beneath it; without one, the archive rides alone,
-    labelled UNSANITIZED. THE FIRST ROUTER CARRIES THE VIM LESSON
-    (2026-09-20): `first` is None to decide by absence, cached per process in
-    _ROUTER_FIRST, or True/False to force it, as the probe does.
+    archive or a preview, and nothing else. A HUMAN'S FILE IS APPENDED TO
+    (2026-09-25, operator's ruling): any other uncommented line means the
+    file is theirs, so this walk lands as one marked block at the bottom,
+    the lines above untouched, and a second write in the same ride replaces
+    only its own block. With a preview, the preview is the line that rides
+    and the archive is a commented line beneath it; without one, the archive
+    rides alone, labelled UNSANITIZED. THE FIRST ROUTER CARRIES THE VIM
```

```
+    LESSON (2026-09-20): `first` is None to decide by absence, cached per
+    process in _ROUTER_FIRST, or True/False to force it, as the probe does.
+    Returns (target, receipt): the path written and one sentence saying how.
     """
     selected = os.environ.get("PIPULATE_ADHOC_FILE", str(REPO_ROOT / "context.txt"))
     if not selected.strip():
@@ -418,18 +421,19 @@ def _write_walk_router(archive_path, preview_path=None,
first=None):
     preview = None if preview_path is None else _router_line(preview_path, "preview")
     # REPLACE ONLY A FILE THIS WRITER COULD HAVE WRITTEN: the introduction line
     # at most, plus one uncommented absolute path to an archive or a preview,
-    # and nothing else. Any other uncommented line is a human's, so the paths
-    # are printed for them instead.
+    # and nothing else. Any other file with lines in it is a human's, and a
+    # human's file is APPENDED TO, never replaced (2026-09-25): one marked
+    # block at the bottom, this walk's own, replaced in place when the same
+    # ride writes twice, so their lines above never move.
+    existing = ""
+    appended = False
     if target.is_file():
-        owned = [line.strip() for line in target.read_text(encoding="utf-8").splitlines()
+        existing = target.read_text(encoding="utf-8")
+        owned = [line.strip() for line in existing.splitlines()
                  if line.strip() and not line.lstrip().startswith("#")]
         routed = [line for line in owned if line != str(INTRO_PATH)]
-        if owned and (len(routed) != 1 or not routed[0].startswith("/")
-                      or not routed[0].endswith(("captures.md", "decant-preview.md"))):
-            raise ValueError(
-                f"{target.name} has lines of your own, so nothing was written; add the line(s) yourself:
"
-                + " ".join(line for line in (preview, source) if line)
-                + " (or delete the file and the next walk rewrites it)")
+        appended = not (owned and len(routed) == 1 and routed[0].startswith("/")
+                        and routed[0].endswith(("captures.md", "decant-preview.md")))
     if first is None:
         if _ROUTER_FIRST["first"] is None:
             _ROUTER_FIRST["first"] = not target.is_file()
@@ -454,7 +458,26 @@ def _write_walk_router(archive_path, preview_path=None,
first=None):
         )
         listing = [line for line in (intro, source) if line]
     body = "\n".join(legend + ["", "# --- BEGIN LISTING FILES OR `! ` COMMANDS ---"] + listing)
+ "\n"
-    body += ROUTER_TAIL if first else "\n# The next walk rewrites this file unless you have
edited it.\n"
+    body += ROUTER_TAIL if first else "\n# Walks rewrite this file until you edit it; after that they
append below.\n"
+    if appended:
```

```
+        run_id = Path(archive_path).parent.name
+        start, end = f"# --- WALK {run_id} START ---", f"# --- WALK {run_id} END ---"
+        # The introduction describes a newcomer; the person whose lines sit
+        # above is not one, so it rides commented here, one keystroke away.
+        block_lines = [start] + legend + ["# --- BEGIN LISTING FILES OR `! ` COMMANDS ---"]
+        block_lines += [f"# {line}" if line == intro else line for line in listing]
+        block = "\n".join(block_lines + [end]) + "\n"
+        at, to = existing.find(start), existing.find(end)
+        if at != -1 and to > at:
+            text = existing[:at] + block + existing[to + len(end):].lstrip("\n")
+            receipt = "replaced this walk's own block at the bottom; your lines above are
untouched"
+        else:
+            text = (existing.rstrip("\n") + "\n\n" if existing.strip() else "") + block
+            receipt = f"appended WALK {run_id} at the bottom; your lines above are untouched"
+    else:
+        text = head + body
+        receipt = ("0600; lists the summary; the archive is a commented line beneath it" if preview
+                   else "0600; lists the UNSANITIZED archive until a summary is saved")
     temp = None
     try:
         with tempfile.NamedTemporaryFile(
@@ -463,7 +486,7 @@ def _write_walk_router(archive_path, preview_path=None, first=None):
         ) as stream:
             temp = Path(stream.name)
             os.fchmod(stream.fileno(), 0o600)
-            stream.write(head + body)
+            stream.write(text)
             stream.flush()
             os.fsync(stream.fileno())
         os.replace(temp, target)
@@ -471,7 +494,7 @@ def _write_walk_router(archive_path, preview_path=None, first=None):
     finally:
         if temp is not None:
             temp.unlink(missing_ok=True)
-    return target
+    return target, receipt


 def _finish_capture_archive(archive, status, skipped=()):
@@ -489,12 +512,12 @@ def _finish_capture_archive(archive, status, skipped=()):
     _print_next_compile(archive["path"])
     if status == "complete" and archive["previews"]:
         try:
-            target = _write_walk_router(archive["path"])
+            target, receipt = _write_walk_router(archive["path"])
         except Exception as exc:
             print(f"  CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
```

```
            print("  Archive preserved; the context file on disk is unchanged.")
        else:
-           print(f"  CONTEXT FILE  {target}  (0600; lists the UNSANITIZED archive until a
summary is saved)")
+           print(f"  CONTEXT FILE  {target}  ({receipt})")
    else:
        print("  CONTEXT FILE unchanged: this run did not complete with captures.")
        print("  An existing context.txt still lists an earlier completed run.")
@@ -693,11 +716,11 @@ def _decant_to_clipboard(payload, archive_path=None):
        # written here, after the file, never inferred from a handoff boolean.
        if archive_path is not None:
            try:
-               router = _write_walk_router(archive_path, preview_path=target)
+               router, receipt = _write_walk_router(archive_path, preview_path=target)
            except Exception as exc:
                print(f"   CONTEXT FILE NOT UPDATED ({type(exc).__name__}): {exc}")
            else:
-               print(f"   CONTEXT FILE  {router}  (0600; lists the summary; the archive is a
commented line beneath it)")
+               print(f"   CONTEXT FILE  {router}  ({receipt})")
    copy_to_clipboard(scrubbed)
    return True
```

```
(nix) pipulate $ m
📝 Committing: chore: Update router_cat.py to improve walk logic and handling of file editing
[main 3f6eff35] chore: Update router_cat.py to improve walk logic and handling of file editing
 1 file changed, 50 insertions(+), 27 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 48 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.52 KiB | 1.52 MiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:pipulate/pipulate.git
   a23b072c..3f6eff35  main -> main
(nix) pipulate $
```

Nice patch there. Ignition is the Nix rebuild:

```
(nix) pipulate $ exit
exit
(sys) pipulate $ ndq
(nix) pipulate $ context
(nix) pipulate $
```

Confirmed, adhoc.txt is renamed to context.txt. Very nice. Less language that needs to be deciphered and more "Oh of course, it couldn't be any other way" intuitively obvious language

that helps me stare down anyone who tries to say anything here is strange or off the beaten track. It's all just text-files. Nothing but text-files doing exactly what they say all built as close as possible to the Unix-way that powers everything under the surface and has for the past 50 years, thank-you-very-much. Your stuff is the fly-by-night weird proprietary stuff with a high churn rate; not mine. This is just generic nothing special everything. There is nothing here that's not standard or done in anything but the most obvious and plain way possible.
4: Prompt: THE AFTER READING for the rename-and-append ride, one probe at a time:

1. The adhoc census is a CENSUS this turn: foo_files.py 6, scripts/git_hooks/pre-commit 1, .gitignore 2, the connectors, GLOSSARY.md 3, the two SKILL.md files, two_arm.py 1 and ascii_displays.py 7, all unchanged.
2. The export: exactly one .env line, now ending in context.txt, and ENV= reading /home/mike/.local/state/pipulate/context.txt; the Processing Log's ROUTER LOADED line names context.txt.
3. The state directory lists context.txt and neither retired name.
4. SOUNDS None None before; two paths under assets/sounds after, with HOME pointed away so the home copies cannot answer for the repo.
5. git ls-files assets/sounds: 0 -> 3 (two wavs and the README).
6. The append fixture: the ValueError line before; APPEND_PROBE blocks=1 mine=1 lines=10 after, meaning two calls left one block, your line survived, and the file is your two lines, a blank, and a seven-line block. A real walk prints "appended WALK walk-... at the bottom" on its first write and "replaced this walk's own block" on its second; ride one whenever you like, and the transcript is the witness.
7. lines-of-your-own census: scripts/mother_cat.py 1 -> 0, foo_files.py 1 both taps; its HANDOFF paragraph says IT REPLACES, IT NEVER MERGES, and the next turn's car adds the append clause, with the file in context.
8. rg -nw cpr prompt_foo.py: two lines, both taps; the 2026-09-18 conviction stays as history, and the other gets a ruling. Car receipts: HEADER_PRUNED with template=40 and tail_muted=12 (three files and nine probe echoes), then RENAMED adhoc.txt->context.txt env_lines_before=2 env_lines_after=1 adhocwalk_removed=True.

NEXT TURN'S CARS: the HANDOFF paragraph and the Selection line in foo_files.py chapter VIII-b (append, never replace, for a human's file); the bunny's new words registered in imports/ascii_displays.py as a three-region edit (render function, CRC ledger, registry) with figurate white_rabbit reading Drift: 0 as the ignition; a ruling on the second cpr line in prompt_foo.py. RULINGS CARRIED: whether the reading frame gets a word back or stays compile --frame lean; whether PIPULATE_ADHOC_FILE, ADHOC_CHOP, the ADHOC SLOT markers and walk.py's adhoc_mutation key are renamed, each a gated car with its readers in context; whether page 3 of the walk names prompt and compile beside its question; the six dated adhoc lines in foo_files.py, for a forget ride; the GLOSSARY.md purge, a ride of its own.
5: Deliverables: Smoooothness.
Oh I didn't even mention the rabbit. Let's just do it...
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ art
(nix) pipulate $ d
diff --git a/imports/ascii_displays.py b/imports/ascii_displays.py

```
index 4a23b370..46495d42 100644
--- a/imports/ascii_displays.py
+++ b/imports/ascii_displays.py
@@ -526,13 +526,13 @@ def _expand_color_bits_ai(text: str) -> str:
 def _figurate_white_rabbit():
     """Render white_rabbit as (human, ai) tuple for FIGURATE_REGISTRY."""
     art = r"""
-                    ( Like a [[[canary]]] you say? )
-                                    O       /)  ____           <debug>The "No Problem"
Framework</debug>
->  I HEREBY WILL NOT RE-GENERATE           o /)\__//  /    \
<success>Pipulate</success> - Protecting Your Code
->  Once upon machines be smarten          ___(/_ 0 0  |     |       just by being honest about text.
->  ASCII sealing immutata art in        *(    ==(_T_)== [[[NPvg]]] |       (If mangled, then AI
drifted.)
->  This here cony if it's broken         \  )   ""\  |     |           https://pipulate.com
->  Smokin gun drift now in token          |__>-\_>_>  \____/                  🥕🥕🥕
+                    ( "The AI said so" is a transcript, not a record. )
+                                    O       /)  ____     This drawing is checksummed.
+>  Same files, same words, same bytes:     o /)\__//  /    \    If it reaches you altered, the
+>  a change you can replay is a change    ___(/_ 0 0  |     |   model rewrote what it was told to
+>  you can put your name behind.        *(    ==(_T_)== NPvg |   copy, and that is the finding.
+>  One diff per turn, a reading before    \  )   ""\  |     |   Verify with nothing installed:
+>  and after, and a digest you can check.  |__>-\_>_>  \____/    python scripts/foo_cartridge.py
foo.zip
     """
     ai_art = _expand_color_bits_ai(art)
     human_art = _expand_color_bits_human(art)
@@ -544,14 +544,14 @@ def _figurate_player_piano():
     # Adjusted with precise trailing spaces to secure an absolute plumb right border on
substitution
     art = r"""

┌──────────────────────────────────────────────────────────────────────────────────────────────────
──────────────┐
- │ ✂️ PLAYER PIANO PROTOCOL — How Chatbots Edit Local Code                 │
+ │ ✂️ PLAYER PIANO PROTOCOL — How Chatbots Edit Local Code                  │

├──────────────────────────────────────────────────────────────────────────────────────────────────
──────────────┤
  │ Target: path/to/file.py                                  │
- │ ___BOX_SEARCH___                                           │
+ │ ___BOX_SEARCH___                                          │
  │ exact old text (character-for-character)                  │
- │ ___BOX_DIVIDER___                                          │
+ │ ___BOX_DIVIDER___                                         │
  │ exact new text                                           │
- │ ___BOX_REPLACE___                                          │
```

```
+ │ ___BOX_REPLACE___                                                    │
  │                                                                      │
  │ 1. Exact match required — no fuzzy edits.                            │
  │ 2. Preserve all whitespace and indentation.                          │
```

```
(nix) pipulate $ m
📝 Committing: chore: Update ascii_displays.py comment text
[main 01ed5274] chore: Update ascii_displays.py comment text
 1 file changed, 11 insertions(+), 11 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 48 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 801 bytes | 801.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:pipulate/pipulate.git
   3f6eff35..01ed5274  main -> main
(nix) pipulate $
```

That's going to throw off some checksums but ironically enough, Fable 5.1 can help me fix it. That ASCII art was designed to be a canary against AIs touching files they weren't supposed to but since I had that premise models like Fable are clearly able to deal with the white spaces in ASCII art. I have to revise that premise and perhaps that messaging too and get its help in getting the checksum back in order which seals the deal on its invalidity as a canary anymore. If a frontier model changed that stuff intentionally you'd never know. If it changed it accidentally it would still throw off checksums so the wax seal functionality still maybe has some validity.
And I think we are getting to the end of this rid. It's an over 6000 line article now and has achieved a massive move forward. Nothing new new please. Let's just tighten and assure it's article wrap-up time. I won't do the backslash K for the article-wrapping protocol until the turn after this one so one more 5-Car Train is welcome.

```
(nix) pipulate $ prompt
(nix) pipulate $ compile
🎨 FIGURATE: DRIFT DETECTED in 'white_rabbit' — expected CRC 3701272927, got 3309949418
```

```
╭──────────────────────────────────────────────────────────────────────────────────────────────────────────────────────╮
──── 🐰 ASCII Art Wax Seal (your vibe-coding safety-net) ─────────────────────────────────────────────────────────────
──────╯
│                                                                                                                      │
│                                ( "The AI said so" is a transcript, not a record. )                                   │
│                                                                                                                      │
│                                                    O       /)  ____     This drawing is checksummed.             │
│  >  Same files, same words, same bytes:      o /)\__//  /    \    If it reaches you altered, the                   │
│                                                                                                                      │
│  >  a change you can replay is a change    ___(/_ 0 0  |     |   model rewrote what it was told to                │
```

```
|
|  >  you can put your name behind.        *(    ==(_T_)== NPvg |   copy, and that is the finding.
|
|  >  One diff per turn, a reading before    \  )   ""\  |      |   Verify with nothing installed:
|
|  >  and after, and a digest you can check.  |__>-\_>_>  \____/    python scripts/foo_cartridge.py
foo.zip                                                          |
|
|
└──────────────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────────────────────
╯
```

🗺️ Codex Mapping Coverage: 74.1% (203/274 tracked files) (was 203/274: +0 claimed, +0 tracked).
ROUTER LOADED: /home/mike/.local/state/pipulate/context.txt (11 active line(s))

✅ Topological Integrity Verified: 45 candidate reference(s) scanned, all exist.
  -> Executing: rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort ... [0.0218s]
  -> Executing: rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate ~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc /etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset} ... [0.0137s]
  -> Executing: ls -1 ~/.local/state/pipulate/ ... [0.0079s]
  -> Executing: HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file; print("SOUNDS", _sound_file("tick.wav"), _sound_file("ding.wav"))' 2>&1 | tail -n 1 ... [0.5670s]
  -> Executing: git ls-files assets/sounds | wc -l ... [0.0107s]
  -> Executing: PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib; sys.path.insert(0,"scripts"); import mother_cat as m; p=pathlib.Path("/tmp/ctxprobe.txt"); p.write_text("# mine\nflake.nix\n"); a=pathlib.Path("/tmp/walk-probe/captures.md"); r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d mine=%d lines=%d" % (t.count("WALK walk-probe START"), t.count("flake.nix"), len(t.splitlines())))' 2>&1 | tail -n 1 ... [0.1632s]
  -> Executing: for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER MERGES' "$f" || echo 0)"; done ... [0.0183s]
  -> Executing: rg -nw cpr prompt_foo.py ... [0.0086s]
  -> Ruff exit 0 (clean).

📦 Payload Ledger (biggest first)

```
┌──────────────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────┬──────────────┬──────────┬──────────
──────────┬──────────────┐
│ File / Source
│  Tokens │     Bytes │ % Bytes │
├──────────────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────────────────────
```

```
──────────────────────────────────────────────────┼──────────────┼──────────
──────┼──────────────┤
│ PROMPT (checklist + prompt.md)
│  99,927 │    380,019 │    33.3% │
│ foo_files.py
│  86,787 │    345,623 │    30.3% │
│ prompt_foo.py
│  49,014 │    215,728 │    18.9% │
│ imports/ascii_displays.py
│  30,399 │    136,609 │    12.0% │
│ apply.py
│  13,316 │     55,136 │     4.8% │
│ .gitignore
│   1,074 │      3,974 │     0.3% │
│ AUTO: Recent Git Diff Telemetry
│     381 │      1,397 │     0.1% │
│ ! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
│     167 │        458 │     0.0% │
│ ! rg -nw cpr prompt_foo.py
│      46 │        163 │     0.0% │
│ ! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile │       43 │        144 │     0.0% │
│ ~/.zshrc /etc/nixos /etc/bashrc /etc/set-environment 2>/dev/null; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}                                 │          │            │          │
│ ! HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file;
print("SOUNDS", _sound_file("tick.wav"), _sound_file("ding.wav"))' │       30 │        104 │     0.0%
│
│ 2>&1 | tail -n 1
│          │            │          │
│ .gitattributes
│      33 │         76 │     0.0% │
│ AUTO: Static Analysis Diagnostics
│      11 │         39 │     0.0% │
│ ! ls -1 ~/.local/state/pipulate/
│      12 │         38 │     0.0% │
│ ! for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER
MERGES' "$f" || echo 0)"; done                    │       13 │         38 │     0.0% │
│ ! PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib;
sys.path.insert(0,"scripts"); import mother_cat as m;             │       13 │         37 │     0.0% │
│ p=pathlib.Path("/tmp/ctxprobe.txt"); p.write_text("# mine\nflake.nix\n");
a=pathlib.Path("/tmp/walk-probe/captures.md");                    │          │            │          │
│ r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d
mine=%d lines=%d" % (t.count("WALK walk-probe START"), │          │            │          │
│ t.count("flake.nix"), len(t.splitlines())))' 2>&1 | tail -n 1
│          │            │          │
│ ! git ls-files assets/sounds | wc -l
│       1 │          1 │     0.0% │
├──────────────────────────────────────────────────┼──────────────┼──────────
──────┼──────────────┤
```

```
┤
│ TOTAL
│ 281,267 │ 1,139,584 │ 100.0% │
└
┘
```

Command: `prompt_foo.py --chop ADHOC_CHOP --no-tree`

--- Auto-Context Metadata ---
• Static Analysis Diagnostics (11 tokens | 39 bytes)
• Recent Git Diff Telemetry (381 tokens | 1,397 bytes)

--- Prompt Summary ---
Summed Tokens: 284,378 (from section parts)
Verified Tokens: 287,715 (from final output)
  (Difference: +3,337)
Total Words: 102,411 (content only)
Total Chars: 1,142,881
Total Bytes: 1,159,652 (UTF-8)

--- Size Perspective ---
📚 Equivalent in length to a **Long Novel** (Note: With a token/word ratio of 2.81, this content is far denser and more complex than typical prose of this length).
🧼 Identity scrub: substitutions=57 rule(s) | denylist=0 rule(s)
🔐 Secrets tripwire: ARMED — 0 hit(s) in payload.
📦 Canonical context cartridge written to /home/mike/repos/pipulate/foo.zip (sha256=b5b500bed195…, members=3)
🗂️ Rotated cartridge snapshot: foo-b5b500be-1565.zip (keeping newest 20 of 20)
Markdown output copied to clipboard
(nix) pipulate $

Note: And the last thought-process we'll capture this article:
Thought for 4m 20s

Reviewing recent receipts to check what's been resolved.

Cross-checking each file's hit counts against predictions.

Tracing a discrepancy in the tail-mute count to a probe timing mismatch.

Tracking a new art drift and untangling its ledger update.

Spotting whitespace edits that undo deliberate alignment padding.

Drafting a script to compare ledger and computed drift values.

Planning to restore the padding exactly to its original bytes.

Weighing whether to splice restored bytes from an earlier commit.

Untangling which commit holds the missing padding bytes.

Verifying the exact CRC value the drift check expects.

Confirming the color markup was stripped, not the seal's job.

Rewriting the handoff comment to reflect append semantics.

Weighing whether claiming an orphan file would tighten coverage.

Drafting the claim line to insert into the capture chapter.

Composing a discharge note marking two owed items resolved.

Untangling a mismatched art region before restoring padded bytes.

Counting box-border characters to verify padding alignment.

Confirming padding choices with rendering-width reasoning before restoring bytes.

Designing probes to detect append markers in the file.

Locating the moved string across a line-number shift.

Explaining what the integrity seal can and cannot detect.

Drafting a safer drift-check command with fallback logging.

Reassessing whether the seal message still fits its true cause.

Drafting the file replacement and reconciling final receipt counts.

Drafting probe commands to verify the discharge and art claims.

Predicting matched checksums to confirm the restored padding.

Fable 5.1 Extra: Context verified. The three cars below patch imports/ascii_displays.py and foo_files.py, both in this payload; the third file a car reaches, the 3f6eff35 blob of ascii_displays.py, comes from git on your machine.

The AFTER reading

1. Census unchanged, a CENSUS as labeled: foo_files.py 6, pre-commit 1, .gitignore 2, the connectors, GLOSSARY.md 3, the two SKILL.md files, two_arm.py 1, ascii_displays.py 7.
2. .env carries one line ending in context.txt, ENV= reads the new path, and the Processing Log's ROUTER LOADED names context.txt with 11 active lines. The rename is done on this machine.
3. The state directory lists context.txt and neither retired name.
4. SOUNDS None None before, both repo paths after, with HOME pointed away.
5. ls-files 0→3.
6. The fixture read the refusal before and blocks=1 mine=1 lines=10 after, all three numbers as predicted: two calls, one block, your line intact.
7. mother_cat.py 1→0. foo_files.py read 0 on both taps against a predicted 1, and the fault is mine twice over: my own Car 10 REPLACE wrapped "IT NEVER" and "MERGES" across two lines, and the probe I wrote for it was a one-line pattern. THE SINGLE-LINE-WITNESS COROLLARY, convicted by the emitter's own block.
8. Both cpr survivors are dated receipts (the 2026-09-18 frame conviction, the 2026-09-21 zero-article gate). Ruled: they stay.

Car receipts: HEADER_PRUNED template=40, tail_muted=11 against a predicted 12, because you had already pasted Context 4 (three files, eight echoes) before running the car and my count used Context 3's nine. RENAMED landed exactly. Your hand git add before m was harmless; m stages with -A and the UNTRACKED footer under d was the receipt that it would. Still unwitnessed: a real walk printing "appended WALK" on its first write and "replaced this walk's own block" on its second. The fixture covered both branches; the ride's own console has not.

The seal

The compile handed over the reseal number itself: expected 3701272927, got 3309949418. Car 2 writes that into the ledger, and the DRIFT DETECTED line vanishing from your next compile is the second witness beside the probe.
The same hand edit changed four lines of player_piano that were not the target. The three ___BOX_ placeholders were padded four columns past the border on purpose: inject_brackets swaps sixteen characters for twelve, so the extra spaces are what makes the rendered box plumb, which is what the comment above that art says. The title line was one space short for the same reason, the scissors emoji being two cells wide in Rich. Car 1 restores those four lines from the 3f6eff35 blob, and the probe's computed equaling the unchanged ledger value 3357002674 is the byte-exact receipt that the restore landed; veto the car if the edit was meant.
On the premise: the seal never knew who touched the painting, only that someone did, which is rung one of THE RECEIPT LADDER and all it ever claimed. Today's ring was your hand. The art's new words attribute an altered copy to a model, a claim the instrument cannot make; softening "the model rewrote" to "something rewrote" is a wording ruling for another turn, not a car tonight.

1. PROBES

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter.

```
rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
```

```
rg -n 'PIPULATE_ADHOC_FILE' ~/repos/pipulate/.env; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
.venv/bin/python -c 'import binascii; from imports.ascii_displays import FIGURATE_REGISTRY
as R, FIGURATE_LEDGER as L; [print("ART", n, "ledger=%d computed=%d" % (L[n],
binascii.crc32(R[n]["render"]()[1].encode("utf-8")))) for n in ("white_rabbit", "player_piano")]'
2>&1 | tail -n 2
rg -n 'WALK <run_id> START' foo_files.py || echo marker=0
rg -n 'assets/sounds/README.md' foo_files.py
rg -c 'DISCHARGED 2026-09-26' foo_files.py || echo 0
```

Lines 1 and 2 are the ride's CENSUS and read the same in both taps. Line 3 straddles Cars 1 and 2: two ART lines, ledger against computed. Line 4 straddles the HANDOFF paragraph. Line 5 is a displacement receipt: the README line moves from the Paintbox to chapter VIII-b, because the next compile regenerates the Paintbox at step 2, before this probe runs, and the Processing Log's Coverage line prints +1 claimed, the positive branch the gauge TODO still lists as unwitnessed. Line 6 straddles the sound TODO.

2. NEXT CONTEXT

```
# --- THE DISMOUNT: foo_files.py is where BANK cars land ---
foo_files.py
# GLOSSARY.md               # <-- uncomment if a graduated key needs its value written
# imports/ascii_displays.py  # <-- only if Car 1 or 2 refused; line 3 is their witness
# scripts/mother_cat.py
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n 'PIPULATE_ADHOC_FILE' ~/repos/pipulate/.env; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
! .venv/bin/python -c 'import binascii; from imports.ascii_displays import FIGURATE_REGISTRY
as R, FIGURATE_LEDGER as L; [print("ART", n, "ledger=%d computed=%d" % (L[n],
binascii.crc32(R[n]["render"]()[1].encode("utf-8")))) for n in ("white_rabbit", "player_piano")]'
2>&1 | tail -n 2
! rg -n 'WALK <run_id> START' foo_files.py || echo marker=0
! rg -n 'assets/sounds/README.md' foo_files.py
! rg -c 'DISCHARGED 2026-09-26' foo_files.py || echo 0
```

3. PATCHES

Three cars. Car 1 is a terminal paste (then d, m). Cars 2 and 3 are patches (patch, app, d, m).

Car 1 (terminal paste): the piano box gets its padding back

Copy the block below, paste it into the terminal where the prompt reads (nix) pipulate $, press Enter, then d, m.

```
python - <<'EOF'
import os, subprocess
```

```
from pathlib import Path
root = Path(os.environ.get("PIPULATE_ROOT", "."))
p = root / "imports" / "ascii_displays.py"
cur = p.read_text(encoding="utf-8").split("\n")
old = subprocess.run(["git", "show", "3f6eff35:imports/ascii_displays.py"], cwd=root,
capture_output=True, text=True, check=True).stdout.split("\n")
def region(t):
    d = next(i for i, l in enumerate(t) if l.startswith("def _figurate_player_piano"))
    s = next(i for i in range(d, len(t)) if t[i].strip() == 'art = r"""')
    e = next(i for i in range(s + 1, len(t)) if t[i].strip() == '"""')
    return s, e
cs, ce = region(cur)
os_, oe = region(old)
assert ce - cs == oe - os_, ("STOP: region length", ce - cs, oe - os_)
changed = [i for i in range(ce - cs + 1) if cur[cs + i] != old[os_ + i]]
assert len(changed) == 4, ("STOP: expected 4 changed lines", changed)
cur[cs:ce + 1] = old[os_:oe + 1]
p.write_text("\n".join(cur), encoding="utf-8")
print(f"PLAYER_PIANO_RESTORED lines {cs + 1}-{ce + 1} restored={len(changed)} from
3f6eff35")
EOF
```

Car 2 (patch): the rabbit's new seal

```
Target: imports/ascii_displays.py
[[[SEARCH]]]
    "white_rabbit": 3701272927,
[[[DIVIDER]]]
    "white_rabbit": 3309949418,
[[[REPLACE]]]
```

Car 3 (patch): foo_files.py catches up with the ride

```
Target: foo_files.py
[[[SEARCH]]]
# THE HANDOFF (97feb327; amended 2026-09-16 and 2026-09-25): a complete,
# nonempty run has the rider write context.txt WHOLE -- the one file
# `context` opens and `compile` reads, PIPULATE_ADHOC_FILE or the repo root
# -- 0600, os.replace'd: the checked preview as the one uncommented line once
# DECANT released it, the whole archive as a commented line beneath, and the
# archive alone, labelled, when no preview was saved. IT REPLACES, IT NEVER
# MERGES, and because `context` invites a hand-typed line into this very
# file, it REPLACES ONLY A FILE IT COULD HAVE WRITTEN: introduction.md plus
# one uncommented absolute path to an archive or a preview and nothing else.
# Any other uncommented line means a human owns the file; the rider prints
# the new path lines and touches nothing, and deleting the file lets the next
```

walk rewrite it (witnessed 2026-09-25 against the operator's own working
list: two refusals, two paths printed, nothing written). SELECTING IS NOT
INCLUDING: naming context.txt inside another list includes its text and
never recurses into its entries.
[[[DIVIDER]]]
THE HANDOFF (97feb327; amended 2026-09-16, 2026-09-25 and 2026-09-26): a
complete, nonempty run has the rider write context.txt -- the one file
`context` opens and `compile` reads, PIPULATE_ADHOC_FILE or the repo root
-- 0600, os.replace'd: the checked preview as the one uncommented line once
DECANT released it, the whole archive as a commented line beneath, and the
archive alone, labelled, when no preview was saved. TWO SHAPES, ONE
WRITER. A file this writer could have written (introduction.md plus one
uncommented absolute path to an archive or a preview, nothing else) is
REPLACED WHOLE, so a newcomer's file never piles up. Any other file with
lines in it is a human's and is APPENDED TO, never replaced (the operator's
ruling of 2026-09-25, landed 2026-09-26): one block at the bottom between
"# --- WALK <run_id> START ---" and its END line, the preview active, the
archive and the introduction commented, and the same ride's second write
replaces only its own block. The 2026-09-25 refusal ("has lines of your
own, nothing written") is gone; the fixture straddle read that refusal
before and blocks=1 mine=1 lines=10 after. SELECTING IS NOT INCLUDING:
naming context.txt inside another list includes its text and never
recurses into its entries.
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
(_capture_checkpoint is a write barrier, not a prompt)
[[[DIVIDER]]]
(_capture_checkpoint is a write barrier, not a prompt)
assets/sounds/README.md   # <-- the tick while the browser opens and the ding when the page has loaded, both CC0 with their sources named; a copy under ~/.local/share/pipulate wins
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
OWED: (1) the Mac, the same two files at the same path, because ~/.local/share never travels; (2) a ruling on whether CC0 files ride in the repo, since the 2026-09-15 ruling against wavs in git predates choosing CC0 for exactly that freedom and *.wav sits in .gitignore's temp block; (3)
[[[DIVIDER]]]
(1) and (2) DISCHARGED 2026-09-26: both files ride in assets/sounds/ beside a README naming their sources, .gitignore carries one negation under its *.wav line, and the loader reads a home copy first and the repo copy second, so the Mac hears them on its next pull (SOUNDS read None None with HOME pointed away, then both repo paths). STILL OWED: (3)
[[[REPLACE]]]

Ignition: none. The probes import the art at call time, and the next compile is what regenerates the Paintbox and prints or omits the DRIFT line.

4. PROMPT

THE AFTER READING for the wrap-up ride, then the dismount:

1. Census and .env are CENSUS lines and read as before.
2. ART white_rabbit ledger=3701272927 computed=3309949418 -> ledger=3309949418 computed=3309949418; ART player_piano ledger=3357002674 computed=(some other number) -> computed=3357002674. Two zeros of drift, and the next compile's console carries no DRIFT DETECTED line. If player_piano's computed still differs after Car 1, the restore was not byte-exact and the honest car is a reseal to the computed value after the rendered box is looked at, never before.
3. marker=0 -> one line in foo_files.py naming WALK <run_id> START.
4. The README line moves from the Paintbox (about line 2915) to chapter VIII-b (about line 1352), and the Processing Log's Coverage reads 204/274 (was 203/274: +1 claimed): the gauge's positive branch, witnessed for the first time; bank it against the gauge TODO in the dismount.
5. DISCHARGED 2026-09-26: 0 -> 1.
Car 1's receipt: PLAYER_PIANO_RESTORED restored=4 from 3f6eff35.

Then \k: the dismount for the context.txt ride. VERIFY against these receipts and the two compiles before them, never memory. BANK: the gauge's +1 reading; the SINGLE-LINE-WITNESS conviction where the emitter's own REPLACE wrapped the phrase its own probe searched for; the seal detects change and never authorship. DANGLING: a real walk's appended block on the console; the art's "the model rewrote" wording; the four remaining rulings (the reading frame's word, the ADHOC-named env var, chop, slot markers and JSON key, page 3 naming prompt and compile, the six dated adhoc lines in foo_files.py for a forget ride, and GLOSSARY.md's purge as its own ride).

5. EXTERNAL DELIVERABLES

None this turn outside the repo.
MikeLev.in: And we fix the drift and the padding.
Same commands, run twice, one change between them. Where the readings differ is what the change did; the diff in the middle is the receipt.
1: Probe:
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
(nix) pipulate $ rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
rg -n 'PIPULATE_ADHOC_FILE' ~/repos/pipulate/.env; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
.venv/bin/python -c 'import binascii; from imports.ascii_displays import FIGURATE_REGISTRY as R, FIGURATE_LEDGER as L; [print("ART", n, "ledger=%d computed=%d" % (L[n], binascii.crc32(R[n]["render"]()[1].encode("utf-8")))) for n in ("white_rabbit", "player_piano")]'

```
2>&1 | tail -n 2
rg -n 'WALK <run_id> START' foo_files.py || echo marker=0
rg -n 'assets/sounds/README.md' foo_files.py
rg -c 'DISCHARGED 2026-09-26' foo_files.py || echo 0
./connectors/botify.py:1
./connectors/confluence.py:1
./connectors/gsc.py:1
./connectors/jira.py:1
./connectors/mcp.py:1
./connectors/README.md:1
./connectors/sheets.py:1
./connectors/slack.py:1
./connectors/wallet.py:3
./foo_files.py:6
./.gitignore:2
./GLOSSARY.md:3
./imports/ascii_displays.py:7
./Notebooks/.agents/skills/gsc_readonly/SKILL.md:1
./Notebooks/.agents/skills/sheets_readonly/SKILL.md:1
./scripts/git_hooks/pre-commit:1
./scripts/two_arm.py:1
12:PIPULATE_ADHOC_FILE="$HOME/.local/state/pipulate/context.txt"
ENV=/home/mike/.local/state/pipulate/context.txt
ART white_rabbit ledger=3701272927 computed=3309949418
ART player_piano ledger=3357002674 computed=1548774637
marker=0
2915:# assets/sounds/README.md  # [195 tokens | 646 bytes]
0
(nix) pipulate $
```

2: Context:

```
# context.txt: the list of files an AI will read. context opens it, compile builds it.
# Keys: j down, k up; Esc then :q quits, :q! discards, :wq saves and quits.
# One line per thing the AI reads: a file path, or a command after `! `.
# A line that starts with # is a comment: that file is not read.
# To add a line: i starts typing, Esc stops. Absolute paths work from anywhere.
# Web pages, APIs and the connector words: chapter XVIII of foo_files.py.

# --- THE 40K-FOOT VIEW (uncomment on a first turn; comment out on the second) ---
# ! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs  # <-- the book's spine, one line per article, newest first
# ~/repos/nixos/autognome.py  # <-- the machine's morning routine (this author's NixOS box only)
# init.lua                    # <-- the editor keys that drive the day
# assets/installer/install.sh # <-- how a stranger's machine gets this workshop
# GLOSSARY.md                 # <-- the terms, defined
# flake.nix                   # <-- the environment, pinned: here is my hardware, here is my state
# prompt_foo.py               # <-- the compiler that builds the payload
```

```
# foo_files.py               # <-- the router: which files ride, and this book's outline
# scripts/articles/lsa.py     # <-- the second brain: the article corpus behind `rgx`, `rgxc` and
`posts`
# requirements.in            # <-- the Python packages, pinned
# pyproject.toml             # <-- the PyPI package
# __init__.py                # <-- the version

# --- ACTUATORS (cheap; include when the AI should be able to act, not only read) ---
# cli.py                    # <-- tool calls from the command line
# scripts/xp.py              # <-- turns a pasted reply into the next context
# scripts/ai.py              # <-- a local AI writes the commit messages
# scripts/crawl.py           # <-- crawl a site into the next turn
# scripts/weblogin.py        # <-- warm a login on the persistent browser profile
# scripts/webclip_2_markdown.py  # <-- a web page, clipped, as markdown

# --- RARE ---
# scripts/foo_cartridge.py    # <-- the sealed archive: writer and verifier
# scripts/foo_replay.py       # <-- replay a sealed archive on another machine
# release.py                 # <-- how a release reaches GitHub and PyPI
# imports/voice_synthesis.py  # <-- the voice
# imports/ascii_displays.py   # <-- the ASCII art
# scripts/release/version_sync.py  # <-- version stamping (to be folded into release.py)

# --- THIS DISCUSSION ---
# The files and commands for the work in front of you. Paste the NEXT CONTEXT
# block an AI hands back directly below this line; the AI will correct a guess.

# Context 1
# # LAUNCH
# walk                       # <-- Repo-root wrapper; delegates to mck.sh, NOT to scripts/walk.py
# assets/installer/mck.sh     # <-- THE LAUNCHER (curl -fsSL pipulate.com/mck.sh | bash):
finds the workshop, forces the spoken rehearsal on first contact, asks for the word RIDE
# scripts/mother_cat.py       # <-- THE RIDER (alias: mothercat), Car B: actuates walk.py's
validated plan; per-run capture banking, DECANT, the adhocwalk.txt writer
# scripts/boot_menu.py        # <-- The one-door command list; short on shell entry, expanded by
`all`, both rendered from tuples in the file
# scripts/sources_menu.py     # <-- The roster `conn` prints; filename stays descriptive, typed
word is shorter
#
# # COMPILE
# scripts/walk_compile.py     # <-- .walk.md -> <name>.yaml BESIDE the surface, and only
where git ignores it; url_env stops only (it refuses a scheme separator anywhere in its output),
so it is the PRIVATE lane's compiler and a bundled trail in assets/trails/ is hand-written JSON;
refuses on every TODO left in the surface; stdlib only, imports nothing
#
# # VALIDATE
# scripts/walk.py             # <-- Car A, the planner: loads a trail, refuses unknown or missing keys
in BOTH directions, prints the dry-run plan; holds DEFAULT_TRAIL; never actuates
```

#
SEAL
scripts/walk_cartridge.py   # <-- trail -> data/walks/<sha256>/walk.zip (trail + manifest with hash and consent surface); reseal is byte-identical
#
CAPTURE
scripts/weblogin.py   # <-- SETTLE ahead of time: warm a login on the persistent profile; --profile default unless told otherwise
tools/scraper_tools.py   # <-- The browser capture and the CAPTURE fence (_capture_checkpoint is a write barrier, not a prompt)
imports/voice_synthesis.py # <-- Piper: scripted narration, never a model (ATTRIBUTED-VOICE)
#
HAND OFF
prompt_foo.py   # <-- Compiles the selected router (adhocwalk.txt: `epr` edits it, `cpr` compiles it) into a payload
scripts/foo_cartridge.py   # <-- The second seal, over the EVIDENCE: payload.md, prompt.md, manifest.json
scripts/foo_replay.py   # <-- Context extraction and attention checks; not browser or API replay
tests/test_mck_rep2.py   # <-- Rep 2: the earmark's owed side-by-side witness
#
THE TRAILS (bundled; each status is the newest receipt, never a promise)
assets/trails/public_walk.json   # profile default; SETTLE trivial; three unlinked npvg.org pages (the_word, the_receipt, the_two_pages), inline script on stop three only, connector noop.py; RIDDEN 2026-09-15 on these stops, 3 of 3 at artifacts=11 each, archive complete, DECANT AUTHORIZED at 20,992 B with checks 0/0/0, and the chatbot given stop three's question said the preview does not carry the server's sentence, which is true (the diff-lens TODO); the 2026-08-01 ride walked the OLD stop set (example.com, mikelev.in, pipulate.com); what bare `walk` rides
assets/trails/practice.yaml   # one configurable page; UNREAD, label stale since 2026-08-01
assets/trails/first_context.yaml   # profile default; SETTLE real; Jira/Botify/Gmail, THREE wallet kinds in one trail; UNRIDDEN
assets/trails/botify_pageworkers.yaml # profile botify; SETTLE real; UNRIDDEN -- the only trail on a non-default profile, and no such ride has ever been witnessed
assets/trails/jira_for_you.yaml   # profile default; SETTLE real (Atlassian sign-in); top of the ticket loop; RIDDEN 2026-09-01 and DECANTED, the sign-in moment unrecorded (a cold profile records it)
assets/trails/ticket.yaml   # profile default; SETTLE real; url_env x2 (JIRA, BOTIFY) fed by one gitignored exports file per ticket under Notebooks/Client_Work/tickets/; stop two of the ticket loop; RIDDEN 2026-09-01 to stop 1 then REFUSED at stop 2 on an unset url_env, Linux and Mac -- the ride that bought the rider its PRE-FLIGHT. RULING: the connector does NOT run at DECANT (DECANT gates only CAPTURE-fenced material; API auth is a different wallet; the rider never harvests). The bridge to the issue text is a ! line in adhoc.txt.
assets/trails/se_ticket.yaml   # profile default; SETTLE real; the SE ticket template: for_you (literal) + issue (JIRA, required) + project/slack/confluence (optional, skipped when unset); RIDDEN 2026-09-02, 3 of 5 captured, 2 skipped, 72,383 B decanted, card showed both

rows; harvest regexes on optional stops are PLACEHOLDERS (botify_analysis's already fullmatched a live analysisSlug); guidance strings hand-count "of five" and go stale on the sixth stop
#
UNLINKED PAGES (npvg.org; nothing links to them and no page links out, so a trail is the only edge; nixops.sh rsyncs them, no rebuild)
remotes/honeybot/www/npvg.org/index.html   # <-- the door: a browser gets this page, curl and wget get install.sh ($npvg_index)
remotes/honeybot/www/npvg.org/walk/1/index.html   # <-- public_walk stop one: the capture word; no script, so source and hydrated DOM should match
remotes/honeybot/www/npvg.org/walk/2/index.html   # <-- stop two: count the archive's fingerprints against the terminal's artifacts= number
remotes/honeybot/www/npvg.org/walk/3/index.html   # <-- stop three: the walk's only script rewrites the server's sentence and appends a paragraph; the DECANT test lives here
#
OFF-ROSTER DISTRIBUTION RESIDUE (not a walk dependency)
assets/installer/replay.sh # <-- OFF the roster 2026-08-01, stranded; re-add needs syntax + one ride + a pinned verifier fetch

Context 2
--- THIS RIDE: the files the cars touched, so the AFTER can read them ---
.gitignore
prompt_foo.py
apply.py
scripts/mother_cat.py
scripts/boot_menu.py
init.lua
flake.nix
foo_files.py
--- NEXT-TURN CARS: the rename's remaining homes; the census names which ---
scripts/xp.py
scripts/git_hooks/pre-commit
introduction.md
scripts/walk.py
scripts/sources_menu.py
GLOSSARY.md
README.md
AGENTS.md
--- THE 40K-FOOT VIEW (your call) ---
! python scripts/articles/lsa.py -t 1 --reverse --fmt dated-slugs
--- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n 'epr\(\) \{|cpr\(\) \{|_walkrouter\(\) \{|context\(\) \{' flake.nix
! .venv/bin/python scripts/boot_menu.py --recall
! rg -n 'with_name\("adhocwalk\.txt"\)|REPO_ROOT / "(adhoc|context)\.txt"|type epr|type context' scripts/mother_cat.py
! rg -n 'pattern = "(adhoc|adhocwalk|context)\.txt"' init.lua
! awk '/^# --- START .(adhoc|context)\.txt. TEMPLATE ---$/{s=NR} /^# --- END

```
.(adhoc|context)\.txt. TEMPLATE ---$/{print "template_lines=" NR-s+1}' foo_files.py
# ! git check-ignore -v context.txt || echo NOT_IGNORED
# ! for f in ~/.config/pipulate/.env ~/.bashrc ~/.profile ~/repos/nixos/configuration.nix; do test -f
"$f" && rg -n 'PIPULATE_ADHOC_FILE' "$f"; done; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
# ! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2

# Context 3
# --- NEXT TURN'S CARS: the rename (once line 2 names the export), the sounds, the walk's
write ---
# scripts/mother_cat.py
# tools/scraper_tools.py
# .gitignore
# # flake.nix        # <-- only if Car 8 or 9 refused; line 7 is their witness
# # foo_files.py     # <-- only if Car 10 refused, or to rule on the six dated adhoc lines it keeps
# # init.lua         # <-- only if Car 6 refused
# # introduction.md  # <-- only if Car 3 refused
# # --- PROBE ECHOES ---
# ! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
# ! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
# ! for f in scripts/git_hooks/pre-commit .git/hooks/pre-commit; do echo "$f:$(rg -c 'context\.txt'
"$f" || echo 0)"; done
# ! rg -cw 'epr|cpr' --sort path prompt_foo.py foo_files.py flake.nix init.lua introduction.md
GLOSSARY.md scripts assets/installer remotes/honeybot/www/npvg.org
# ! for f in scripts/walk.py init.lua scripts/mother_cat.py; do echo "$f:$(rg -c adhoc "$f" || echo 0)";
done
# ! rg -n 'door 2' scripts/sources_menu.py || echo door2=0
# ! LD_LIBRARY_PATH="" nix eval --raw .#devShells.x86_64-linux.quiet.shellHook 2>/dev/null |
rg -c 'Before conducting experiment' || echo 0
# ! nvim --headless --clean -c 'luafile init.lua' -c 'qa!' 2>&1 | tail -n 2
# ! ls -la ~/.local/share/pipulate/; rg -n 'local/share/pipulate|tick\.wav|ding\.wav|\*\.wav'
tools/scraper_tools.py .gitignore

# Context 4
# --- NEXT TURN'S CARS: the HANDOFF paragraph, the bunny, the cpr survivors ---
# foo_files.py
# imports/ascii_displays.py
# prompt_foo.py
# # scripts/mother_cat.py    # <-- only if Car 5 refused; line 6 is its witness
# # tools/scraper_tools.py   # <-- only if Car 3 refused; line 4 is its witness
# # .gitignore
# # --- PROBE ECHOES ---
# ! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
# ! rg -n --hidden --glob '!.git' 'PIPULATE_ADHOC_FILE' ~/repos/nixos ~/.config/pipulate
~/repos/pipulate/.env ~/.bashrc ~/.bash_profile ~/.profile ~/.zshrc /etc/nixos /etc/bashrc
/etc/set-environment 2>/dev/null; echo ENV=${PIPULATE_ADHOC_FILE:-unset}
```

```
# ! ls -1 ~/.local/state/pipulate/
# ! HOME=/tmp .venv/bin/python -c 'from tools.scraper_tools import _sound_file;
print("SOUNDS", _sound_file("tick.wav"), _sound_file("ding.wav"))' 2>&1 | tail -n 1
# ! git ls-files assets/sounds | wc -l
# ! PIPULATE_ADHOC_FILE=/tmp/ctxprobe.txt .venv/bin/python -c 'import sys,pathlib;
sys.path.insert(0,"scripts"); import mother_cat as m; p=pathlib.Path("/tmp/ctxprobe.txt");
p.write_text("# mine\nflake.nix\n"); a=pathlib.Path("/tmp/walk-probe/captures.md");
r=[m._write_walk_router(a) for _ in (1,2)]; t=p.read_text(); print("APPEND_PROBE blocks=%d
mine=%d lines=%d" % (t.count("WALK walk-probe START"), t.count("flake.nix"),
len(t.splitlines())))' 2>&1 | tail -n 1
# ! for f in scripts/mother_cat.py foo_files.py; do echo "$f:$(rg -c 'lines of your own|NEVER
MERGES' "$f" || echo 0)"; done
# ! rg -nw cpr prompt_foo.py

# Context 5
# --- THE DISMOUNT: foo_files.py is where BANK cars land ---
foo_files.py
# GLOSSARY.md               # <-- uncomment if a graduated key needs its value written
# imports/ascii_displays.py  # <-- only if Car 1 or 2 refused; line 3 is their witness
# scripts/mother_cat.py
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! rg -n 'PIPULATE_ADHOC_FILE' ~/repos/pipulate/.env; echo
ENV=${PIPULATE_ADHOC_FILE:-unset}
! .venv/bin/python -c 'import binascii; from imports.ascii_displays import FIGURATE_REGISTRY
as R, FIGURATE_LEDGER as L; [print("ART", n, "ledger=%d computed=%d" % (L[n],
binascii.crc32(R[n]["render"]()[1].encode("utf-8")))) for n in ("white_rabbit", "player_piano")]'
2>&1 | tail -n 2
! rg -n 'WALK <run_id> START' foo_files.py || echo marker=0
! rg -n 'assets/sounds/README.md' foo_files.py
! rg -c 'DISCHARGED 2026-09-26' foo_files.py || echo 0
```

3: Patches:
Left-hand causal boundary "blast radius" established. Make 1 change and test.
On branch main
Your branch is up to date with 'origin/main'.

```
nothing to commit, working tree clean
(nix) pipulate $ context
(nix) pipulate $ python - <<'EOF'
import os, subprocess
from pathlib import Path
root = Path(os.environ.get("PIPULATE_ROOT", "."))
p = root / "imports" / "ascii_displays.py"
cur = p.read_text(encoding="utf-8").split("\n")
old = subprocess.run(["git", "show", "3f6eff35:imports/ascii_displays.py"], cwd=root,
capture_output=True, text=True, check=True).stdout.split("\n")
def region(t):
```

```
    d = next(i for i, l in enumerate(t) if l.startswith("def _figurate_player_piano"))
    s = next(i for i in range(d, len(t)) if t[i].strip() == 'art = r"""')
    e = next(i for i in range(s + 1, len(t)) if t[i].strip() == '"""')
    return s, e
cs, ce = region(cur)
os_, oe = region(old)
assert ce - cs == oe - os_, ("STOP: region length", ce - cs, oe - os_)
changed = [i for i in range(ce - cs + 1) if cur[cs + i] != old[os_ + i]]
assert len(changed) == 4, ("STOP: expected 4 changed lines", changed)
cur[cs:ce + 1] = old[os_:oe + 1]
p.write_text("\n".join(cur), encoding="utf-8")
print(f"PLAYER_PIANO_RESTORED lines {cs + 1}-{ce + 1} restored={len(changed)} from
3f6eff35")
EOF
PLAYER_PIANO_RESTORED lines 545-561 restored=4 from 3f6eff35
(nix) pipulate $ d
diff --git a/imports/ascii_displays.py b/imports/ascii_displays.py
index 46495d42..207a5a23 100644
--- a/imports/ascii_displays.py
+++ b/imports/ascii_displays.py
@@ -544,14 +544,14 @@ def _figurate_player_piano():
     # Adjusted with precise trailing spaces to secure an absolute plumb right border on
substitution
     art = r"""

┌──────────────────────────────────────────────────────────────────────────────────────
──────────────────┐
- │ ✂️ PLAYER PIANO PROTOCOL — How Chatbots Edit Local Code                │
+ │ ✂️ PLAYER PIANO PROTOCOL — How Chatbots Edit Local Code                │

├──────────────────────────────────────────────────────────────────────────────────────
──────────────────┤
  │ Target: path/to/file.py                                                │
- │ ___BOX_SEARCH___                                                       │
+ │ ___BOX_SEARCH___                                                       │
  │ exact old text (character-for-character)                               │
- │ ___BOX_DIVIDER___                                                      │
+ │ ___BOX_DIVIDER___                                                      │
  │ exact new text                                                         │
- │ ___BOX_REPLACE___                                                      │
+ │ ___BOX_REPLACE___                                                      │
  │                                                                        │
  │ 1. Exact match required — no fuzzy edits.                              │
  │ 2. Preserve all whitespace and indentation.                            │
(nix) pipulate $ m
📝 Committing: chore: Update placeholder text in ascii_displays.py
[main 5b6714e7] chore: Update placeholder text in ascii_displays.py
 1 file changed, 4 insertions(+), 4 deletions(-)
```

```
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'imports/ascii_displays.py'.
(nix) pipulate $ d
diff --git a/imports/ascii_displays.py b/imports/ascii_displays.py
index 207a5a23..d655ead7 100644
--- a/imports/ascii_displays.py
+++ b/imports/ascii_displays.py
@@ -53,7 +53,7 @@ console = Console()
 # To add a new entry: print(binascii.crc32(your_art_string.encode('utf-8')))

 FIGURATE_LEDGER: dict = {
-    "white_rabbit": 3701272927,
+    "white_rabbit": 3309949418,
     "player_piano": 3357002674,
     "clipboard": 858911667,
     "bunny_trail": 615479347,
(nix) pipulate $ m
📝 Committing: chore: Update FIGURATE_LEDGER constants
[main af5b0847] chore: Update FIGURATE_LEDGER constants
 1 file changed, 1 insertion(+), 1 deletion(-)
(nix) pipulate $ patch
(nix) pipulate $ app
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
✅ DETERMINISTIC PATCH APPLIED: Successfully mutated 'foo_files.py'.
(nix) pipulate $ d
diff --git a/foo_files.py b/foo_files.py
index ab849754..22ddc594 100644
--- a/foo_files.py
+++ b/foo_files.py
@@ -1351,6 +1351,7 @@ foo_files.py      #  <-- THIS file. Content compiler router. Makes it
very meta.
 # CAPTURE
 # scripts/weblogin.py        # <-- SETTLE ahead of time: warm a login on the persistent profile;
--profile default unless told otherwise
 # tools/scraper_tools.py     # <-- The browser capture and the CAPTURE fence
(_capture_checkpoint is a write barrier, not a prompt)
+# assets/sounds/README.md     # <-- the tick while the browser opens and the ding when the
page has loaded, both CC0 with their sources named; a copy under ~/.local/share/pipulate wins
 # imports/voice_synthesis.py  # <-- Piper: scripted narration, never a model
(ATTRIBUTED-VOICE)
 #
 # HAND OFF
@@ -1386,21 +1387,24 @@ foo_files.py      #  <-- THIS file. Content compiler router. Makes it
very meta.
 # Selection: context.txt names the checked preview and, commented beneath it, the raw
captures.md;
```

 # the list is neither disclosure approval nor the evidence.
 #
-# THE HANDOFF (97feb327; amended 2026-09-16 and 2026-09-25): a complete,
-# nonempty run has the rider write context.txt WHOLE -- the one file
+# THE HANDOFF (97feb327; amended 2026-09-16, 2026-09-25 and 2026-09-26): a
+# complete, nonempty run has the rider write context.txt -- the one file
 # `context` opens and `compile` reads, PIPULATE_ADHOC_FILE or the repo root
 # -- 0600, os.replace'd: the checked preview as the one uncommented line once
 # DECANT released it, the whole archive as a commented line beneath, and the
-# archive alone, labelled, when no preview was saved. IT REPLACES, IT NEVER
-# MERGES, and because `context` invites a hand-typed line into this very
-# file, it REPLACES ONLY A FILE IT COULD HAVE WRITTEN: introduction.md plus
-# one uncommented absolute path to an archive or a preview and nothing else.
-# Any other uncommented line means a human owns the file; the rider prints
-# the new path lines and touches nothing, and deleting the file lets the next
-# walk rewrite it (witnessed 2026-09-25 against the operator's own working
-# list: two refusals, two paths printed, nothing written). SELECTING IS NOT
-# INCLUDING: naming context.txt inside another list includes its text and
-# never recurses into its entries.
+# archive alone, labelled, when no preview was saved. TWO SHAPES, ONE
+# WRITER. A file this writer could have written (introduction.md plus one
+# uncommented absolute path to an archive or a preview, nothing else) is
+# REPLACED WHOLE, so a newcomer's file never piles up. Any other file with
+# lines in it is a human's and is APPENDED TO, never replaced (the operator's
+# ruling of 2026-09-25, landed 2026-09-26): one block at the bottom between
+# "# --- WALK <run_id> START ---" and its END line, the preview active, the
+# archive and the introduction commented, and the same ride's second write
+# replaces only its own block. The 2026-09-25 refusal ("has lines of your
+# own, nothing written") is gone; the fixture straddle read that refusal
+# before and blocks=1 mine=1 lines=10 after. SELECTING IS NOT INCLUDING:
+# naming context.txt inside another list includes its text and never
+# recurses into its entries.
 #
 # AUTH RULING (banked 2026-08-09, source-witnessed): THERE IS NO AUTH FIELD IN
 # THE TRAIL SCHEMA. walk.py enforces set-difference in BOTH directions over the
@@ -2196,7 +2200,7 @@ MATCHBOOK_CHOP = r"""
 # - TODO (2026-09-15): one public_walk ride left 6 non-curl /walk/ lines in npvg.access.log, two per page; the user-agent split (deed 1408) read ONE agent, Chrome/150 on X11, with one 200 and one 304 per page, so there is no second fetcher: the same browser loaded each page twice and the second load was a conditional revalidation. Read the six lines in log order (time, status, path, no address) to rule between a double load inside one stop and a restore of the previous tab when the next stop's browser launches; count no rides until that is ruled, and filter the operator's own rides by address before reading the funnel at all.

-# - TODO (2026-09-15, operator ruling; mechanism landed with THE ONE DOOR; FILES LANDED 2026-09-19 ON THE LINUX BOX, deed 1495): tick.wav is OpenGameArt node 16323 (AntumDeluge, CC0; clock-1.wav, 3.66 s) and ding.wav is Freesound 611113 (5ro4, CC0; a

pet-training bell, 2.52 s), both re-encoded pcm_s16le 44100 stereo by ffmpeg into ~/.local/share/pipulate/ and heard by ear at the prompt, balanced. The scraper loops the tick in its own process group (tools/scraper_tools.py:97, setsid) from before uc.Chrome until driver.get returns, SIGKILLs the group and plays the ding once (:115, called with ding=True at :659), and stays silent when the files are absent. OWED: (1) the Mac, the same two files at the same path, because ~/.local/share never travels; (2) a ruling on whether CC0 files ride in the repo, since the 2026-09-15 ruling against wavs in git predates choosing CC0 for exactly that freedom and *.wav sits in .gitignore's temp block; (3) a ruling on whether the sounds follow the voice consent (~/.config/pipulate/voice, can_speak() in imports/voice_synthesis.py) or stay file-presence-gated as today; (4) a console line when the loop starts and when the ding fires, because the sound path prints nothing and silence is three worlds (no files, no player, a loop that died at spawn). The tick and ding on a real walk were witnessed by ear on 2026-09-19 (deed 1496, walk-wb2bqtax, three stops at artifacts=12): the tick looped from "Opening the browser" until "The page is open", one ding fired before each CAPTURE prompt, and the console printed nothing about sound, exactly the silence owed at (4). The settle was far faster than the 2026-09-15 walk because the guided lane skips the 20 s staleness timeout (tools/scraper_tools.py, the `if not interactive:` guard read at deed 1495) and the ding lands the moment driver.get returns at the page's load event; after that the human's eye is the settle detector, which is what the CAPTURE fence is for.
+# - TODO (2026-09-15, operator ruling; mechanism landed with THE ONE DOOR; FILES LANDED 2026-09-19 ON THE LINUX BOX, deed 1495): tick.wav is OpenGameArt node 16323 (AntumDeluge, CC0; clock-1.wav, 3.66 s) and ding.wav is Freesound 611113 (5ro4, CC0; a pet-training bell, 2.52 s), both re-encoded pcm_s16le 44100 stereo by ffmpeg into ~/.local/share/pipulate/ and heard by ear at the prompt, balanced. The scraper loops the tick in its own process group (tools/scraper_tools.py:97, setsid) from before uc.Chrome until driver.get returns, SIGKILLs the group and plays the ding once (:115, called with ding=True at :659), and stays silent when the files are absent. (1) and (2) DISCHARGED 2026-09-26: both files ride in assets/sounds/ beside a README naming their sources, .gitignore carries one negation under its *.wav line, and the loader reads a home copy first and the repo copy second, so the Mac hears them on its next pull (SOUNDS read None None with HOME pointed away, then both repo paths). STILL OWED: (3) a ruling on whether the sounds follow the voice consent (~/.config/pipulate/voice, can_speak() in imports/voice_synthesis.py) or stay file-presence-gated as today; (4) a console line when the loop starts and when the ding fires, because the sound path prints nothing and silence is three worlds (no files, no player, a loop that died at spawn). The tick and ding on a real walk were witnessed by ear on 2026-09-19 (deed 1496, walk-wb2bqtax, three stops at artifacts=12): the tick looped from "Opening the browser" until "The page is open", one ding fired before each CAPTURE prompt, and the console printed nothing about sound, exactly the silence owed at (4). The settle was far faster than the 2026-09-15 walk because the guided lane skips the 20 s staleness timeout (tools/scraper_tools.py, the `if not interactive:` guard read at deed 1495) and the ding lands the moment driver.get returns at the page's load event; after that the human's eye is the settle detector, which is what the CAPTURE fence is for.
 # - TODO (2026-09-13): npvg.org rides mikelev.in's public address as a STATIC A record while only mikelev.in gets the namecheap-ddns heartbeat; a second unit (domain=npvg.org, its own token) or an ALIAS record is owed before the next public-IP change, or the pad goes dark with nothing printed anywhere.
 # - TODO (2026-09-13): the pad has no favicon, so every browser visit writes a 404 line into npvg.access.log beside its 200 (the cellular witness was two lines for one visit); a locations."=

/favicon.ico" returning 204 is three lines, owed before the funnel is read from that log.
 # - EARMARK: THE DIRECTORY KEY CROSSED MACHINES (banked 2026-09-08, convicted by the second machine): a shared file may not name a thing by a MACHINE-LOCAL handle, because the handle is assigned per machine and the file is not. Chrome numbers profile directories in creation order, so "Profile 2" was the Work profile on NixOS and the PERSONAL profile on the Mac (Mike, 121 synced bookmarks), and bookmarks.nix, keyed by that directory, resolved to the wrong profile the first time a second checkout ran it; only the sync_metadata fence stood between a real run and a harvest-and-wipe of a personal bar. The key had been right for the whole life of the feature on the one machine that existed, which is SINGLE-CANDIDATE BLINDNESS wearing a filename: n=1 cannot tell a stable identity from a coincidence. CURE, a grammar change rather than a value change: the key is a human name and a `match` list names IDENTITIES the machine derives and the operator cannot author (a label, an account, an account domain, read from the application's own registry); exactly one hit resolves, none skips, several refuse. THE TEST before any key rides a shared file: is this value assigned by the machine in an order the other machine never saw? Directory numbers, pids, device paths, X display numbers and "Profile N" all fail it. Sibling of THE DERIVED-PATH RULE (a write target computed from an identity the writer cannot author) and of SINGLE-CANDIDATE BLINDNESS (create the second candidate); this names the class of key the second candidate convicts.
(nix) pipulate $ m
📝 Committing: chore: Add README.md for browser sounds
[main 5dca1c44] chore: Add README.md for browser sounds
 1 file changed, 17 insertions(+), 13 deletions(-)
(nix) pipulate $ git push
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 48 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 1.91 KiB | 391.00 KiB/s, done.
Total 11 (delta 8), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (8/8), completed with 4 local objects.
To github.com:pipulate/pipulate.git
   3a8b5c91..5dca1c44 main -> main
(nix) pipulate $

Okay we're almost to the end of this most epic article. I have reason to think the human can still see ASCII art in ways LLMs can't given that it reverted that art instead of updating its checksum recognizing the "lining up" I was doing.
No ignition. We're done.
4: Prompt: THE AFTER READING for the wrap-up ride, then the dismount:

1. Census and .env are CENSUS lines and read as before.
2. ART white_rabbit ledger=3701272927 computed=3309949418 -> ledger=3309949418 computed=3309949418; ART player_piano ledger=3357002674 computed=(some other number) -> computed=3357002674. Two zeros of drift, and the next compile's console carries no DRIFT DETECTED line. If player_piano's computed still differs after Car 1, the restore was not byte-exact and the honest car is a reseal to the computed value after the rendered box is looked at, never before.
3. marker=0 -> one line in foo_files.py naming WALK START.

4. The README line moves from the Paintbox (about line 2915) to chapter VIII-b (about line 1352), and the Processing Log's Coverage reads 204/274 (was 203/274: +1 claimed): the gauge's positive branch, witnessed for the first time; bank it against the gauge TODO in the dismount.
5. DISCHARGED 2026-09-26: 0 -> 1. Car 1's receipt: PLAYER_PIANO_RESTORED restored=4 from 3f6eff35.

Then \k: the dismount for the context.txt ride. VERIFY against these receipts and the two compiles before them, never memory. BANK: the gauge's +1 reading; the SINGLE-LINE-WITNESS conviction where the emitter's own REPLACE wrapped the phrase its own probe searched for; the seal detects change and never authorship. DANGLING: a real walk's appended block on the console; the art's "the model rewrote" wording; the four remaining rulings (the reading frame's word, the ADHOC-named env var, chop, slot markers and JSON key, page 3 naming prompt and compile, the six dated adhoc lines in foo_files.py for a forget ride, and GLOSSARY.md's purge as its own ride).

5: Deliverables: A vastly less objectionable experience in terms of terms.

Hop off the ride. This ride's stated goal is reached — dismount. This is the NOTARY BEAT: the ride ends here, is witnessed here, and is sealed here. Answer all seven beats, briefly:

0. **TL;DR**: a short, dry, neutral abstract for the TOP of the published article — written for an unfamiliar reader or AI summarizer who has never seen this system. No hype, no insider handles unexplained.
1. VERIFY: restate the goal from the top of this article and confirm (or deny) it was met, citing THIS compile's receipts, not memory. Name any ignition this ride required that never fired -- an AFTER tap taken without one is a stale BEFORE wearing the AFTER's label.
2. BANK: name everything that graduates — rule, earmark, todo, pin — as SEARCH/REPLACE patch cars against raw source present in THIS context (Target line inside the fence, one car per commit story), deletions included. A 'paste-ready' line with no Target and no SEARCH anchor is not banked; it is a hand edit the operator will not make. These BANK cars are the ONLY patches a dismount emits.
3. DANGLING: what carries forward unbanked? One line each, no essays.
4. SEED: the context.txt lines (and TODO_SLUGS if narrative context is needed) for the next ride's first compile.
5. CLOSING: a closing summary for the BOTTOM of the article — the final take-away, tied to the book's larger arc where it fits naturally, never forced. Storytelling over inventory.
6. NOTARIZE: the artifact of record is the cartridge THIS compile writes after this payload is assembled, so its name is not in your context and you must not guess one. Attest the FINGERPRINT instead: this Manifest's wc -c line, its receipt count, and the newest receipt's date -- pure functions of the bytes the cartridge seals. Close with one line the operator completes after the compile: Deed: foo-__.zip (ls -t foo-*.zip | head -1) The seal is the signature; the fingerprint is how the deed is found.

FINALITY: after beat 6, this discussion is CLOSED. Emit NO five-car train, NO probes, NO patches beyond the BANK cars of beat 2, and NO next-turn prompt beyond the SEED lines in beat 4. Any reader or model encountering this article later should treat it as a finished, notarized document — an archive entry, not an open thread.

Fable 5.1 Extra:

1. VERIFY

The goal at the top: one file where there were two, three words where there were four, nothing on a newcomer's screen that needs a glossary, and, added mid-ride, the sounds in the repo, the walk appending to a human's file, the rabbit reworded. Met, on this compile's receipts:

- ! rg -n ... PIPULATE_ADHOC_FILE: .env:12 names context.txt, ENV= reads the same path, and the Processing Log's ROUTER LOADED line names context.txt with 11 active lines.
- ! ls -1 ~/.local/state/pipulate/: context.txt, and neither retired name.
- ! HOME=/tmp ... SOUNDS: both repo paths with the home copies out of reach; ! git ls-files assets/sounds | wc -l: 3.
- ! ... APPEND_PROBE: blocks=1 mine=1 lines=10, the three numbers as predicted.
- ! for f ... 'lines of your own|NEVER MERGES': 0 and 0; the HANDOFF paragraph in this payload's foo_files.py reads TWO SHAPES, ONE WRITER and names the WALK <run_id> START marker.
- ! rg -nw cpr prompt_foo.py: two dated lines (1791, 2394); ruled to stay.
- ! rg -c ... 'adhoc(walk)?\.txt': unchanged from the prior tap, a CENSUS as labeled; every code home of the old name reads absent.
- Processing Log: Coverage: 74.5% (204/274 tracked files) (was 203/274: +1 claimed, +0 tracked). The gauge's positive branch, printed for the first time, by an ordinary claim rather than the two-fence choreography. Telemetry carries that regeneration uncommitted in the working tree; the next m sweeps it.
- Codebase bodies: ascii_displays.py carries "white_rabbit": 3309949418, the number the 1565 compile printed as computed, and the piano's four lines carry their padding again; foo_files.py's sound TODO reads (1) and (2) DISCHARGED 2026-09-26 and assets/sounds/README.md sits in chapter VIII-b.

THE STALE OVERLAY, said out loud: the pasted context.txt shows Context 5 active, and the Manifest ran Context 4's eight probes. So the ART crc, the WALK marker, the README displacement and the DISCHARGED count never ran in this compile. The file bodies above stand in for three of them. The two zeros of drift are INFERRED: white_rabbit from the ledger now matching the compile's own computed value, player_piano from restored=4 out of the 3f6eff35 blob the ledger sealed. The console's DRIFT line is a logging warning that never enters the Processing Log, so its absence is yours to read, not mine.

Ignition fired where required (exit then ndq, twice). Never fired: a real walk after the append landed, so "appended WALK" is a fixture's word and not yet the console's; and the Mac has pulled none of this.

On the piano: I did read the lining up. The comment above that art says the padding is for the render, because inject_brackets trades sixteen characters for twelve, and the source is meant to look wrong so the box comes out plumb; the title line's scissors count as two cells in Rich. Whether the box is plumb on your screen is the eye's ruling, figurate player_piano, and a reseal follows the eye, never me.

2. BANK

Five cars, each its own commit (patch, app, d, m).

Car 1: the ride's receipt

Target: foo_files.py
[[[SEARCH]]]
--- START RECEIPTS (newest first; cap 20 lines; a line pushed past the cap is deleted, never moved -- git and the rolling pin are the archive) ---
[[[DIVIDER]]]
--- START RECEIPTS (newest first; cap 20 lines; a line pushed past the cap is deleted, never moved -- git and the rolling pin are the archive) ---
2026-09-26 dismount THE FILE IS CALLED WHAT IT IS (deed 1567 seals the dismount, 1565 the compile before it; commits 65ae83fc through 5dca1c44, about thirty, one fence each, pushed by hand): adhoc.txt and adhocwalk.txt became one context.txt, and epr, cpr and _walkrouter left the flake for context, prompt and compile on both menu lists, each newcomer-facing string rewritten in plain words. Straddles in band: .env 2 lines -> 1 naming context.txt with ENV= agreeing; the state dir listing context.txt and neither retired name; the adhoc census prompt_foo.py 14 -> absent, apply.py 3 -> absent, mother_cat.py 4 -> absent, init.lua 8 -> absent, flake.nix 14 -> absent, foo_files.py 22 -> 6 (all dated), pre-commit 3 -> 1 (the for-loop naming all three), the template 125 -> 43 lines; SOUNDS None None -> both repo paths with HOME pointed away, git ls-files 0 -> 3; the append fixture the refusal -> blocks=1 mine=1 lines=10; the Coverage gauge's +1 branch printed for the first time (203 -> 204, the sounds README claimed into VIII-b). The rabbit's words changed by hand and the seal read drift, 3701272927 against 3309949418, resealed to the number the compile printed; the same hand moved four piano lines and a car put them back from 3f6eff35 (restored=4). Misses mine: INIT_LUA_MERGED 25 old against 24, the template 125 against "about 110", tail_muted 11 against 12 with the paste already landed, and one probe reading 0 in both worlds because my own REPLACE had wrapped its phrase. THE STALE OVERLAY at the dismount: the paste showed Context 5 active and the Manifest ran Context 4, so the ART crc, WALK marker and DISCHARGED probes never ran and the file bodies stood in. UNWITNESSED: a real walk's "appended WALK" on the console, the Mac, the piano box by eye. This block reads one line past the 09-24 reading; the forget ride's awk is the count that rules.
[[[REPLACE]]]

Car 2: the gauge TODO records its +1

Target: foo_files.py
[[[SEARCH]]]
Delete this line when -1 and +1 have printed.
[[[DIVIDER]]]
THE +1 PRINTED 2026-09-26 (deed 1567): a real claim, assets/sounds/README.md moved from the Paintbox into chapter VIII-b, and the Processing Log read 204/274 (was 203/274: +1 claimed, +0 tracked), the positive branch by an ordinary edit rather than the two-fence choreography. Only the -1 remains, and the Car 1 unclaim-and-compile fence above is its shape. Delete this line when -1 has printed.
[[[REPLACE]]]

Car 3: SINGLE-LINE-WITNESS takes its second conviction

Target: foo_files.py
[[[SEARCH]]]
confirm the phrase is unsplit before recommending the probe.
[[[DIVIDER]]]
confirm the phrase is unsplit before recommending the probe. Second conviction 2026-09-26, by the emitter's own hand: a REPLACE block wrapped IT NEVER MERGES across two lines and the same turn's probe searched for it on one, so the reading was 0 in both worlds and the diff was the only witness.
[[[REPLACE]]]

Car 4: rung one names no hand

Target: foo_files.py
[[[SEARCH]]]
handed to a model that cannot see .git.
[[[DIVIDER]]]
handed to a model that cannot see .git.
RUNG ONE NAMES NO HAND (banked 2026-09-26): the rabbit's words were changed
by hand, the wax seal read drift, and it would have read the same drift had
a model changed them; a CRC says "changed" and never "who". A caption that
names the author ("the model rewrote what it was told to copy") is a verdict
written into the instrument. Let the seal say what it measures.
[[[REPLACE]]]

Car 5: the append ruling reaches three 2026-09-16 TODOs

Target: foo_files.py
[[[SEARCH]]]
- TODO (2026-09-16, the walk seam): one real walk after commit 6b10c419 so `cat "$(_walkrouter)"` reads the preview uncommented and the archive commented beneath it; the writer's three branches passed the compile-lane probe, the rider handing it the preview has never run. The Mac is that witness if this machine does not get there first, and needs two nix develop entries, the first pulling these commits and the second running the hook it pulled. MAC READING 2026-09-16: the router does not exist there at all (cat: no such file), because PIPULATE_ADHOC_FILE is unset on that machine and the derivation lands $PIPULATE_ROOT/adhocwalk.txt, gitignored, never written. So the seam is unwitnessed on BOTH machines and the Mac's reading is absence, not staleness -- and `cpr` REFUSES there until a walk or an `epr` write creates the file, which is the fresh-install path a newcomer takes.
- TODO (2026-09-16, the question's home): bare cpr reads prompt.md and a newcomer has none; `cpr "question"` works today and costs quoting; the -o split was tried and reverted. Candidates, none chosen: the walk seeds a prompt.md at DECANT, or the router's header teaches the positional. First rule from prompt_foo.py:2843 whether bare cpr with no prompt.md refuses or proceeds (CENSUS: rg -n -A6 'elif os.path.exists\("prompt.md"\)' prompt_foo.py);

CENSUS RAN 2026-09-16 (Mac compile): prompt_foo.py:2843 is an elif with NO else -- 2845 blank, 2846 dedents to extra_prompt_parts -- so bare cpr with no prompt.md PROCEEDS and no car is owed for a refusal. STILL OPEN, and six lines cannot see it: proceeds with WHAT, since prompt_content's initializer sits above the window (probe: rg -n -B8). An empty Prompt section is worse than a refusal, because nothing announces it. SEPARATE AND SHARPER: cpr refuses when the ROUTER is missing, which is every fresh install before its first walk.
[[[DIVIDER]]]
- TODO (2026-09-16, the walk seam; rewritten 2026-09-26): _walkrouter, epr and cpr are gone, the walk writes the one context.txt, and a human's file is appended to (the fixture read the refusal, then blocks=1 mine=1 lines=10 at deed 1567). STILL OWED: one real walk's console reading "appended WALK <run_id> at the bottom" and, after DECANT, "replaced this walk's own block"; and a fresh install's first walk writing the file whole with the vim lesson, which the Mac is the witness for. The question's home is answered: bare compile with no prompt.md proceeds with the one-line default prompt (prompt_foo.py main, read at deed 1567), compile refuses a missing file only when PIPULATE_ADHOC_FILE names one, which a fresh install does not, and the walk's tail now says type prompt.
[[[REPLACE]]]
Target: foo_files.py
[[[SEARCH]]]
- TODO (2026-09-16, the router's own drill-down): _write_walk_router's header promises "an edited router is never replaced", and its guard keeps any file whose uncommented lines are not exactly one absolute path ending captures.md or decant-preview.md. So the ONE edit the header invites -- uncommenting the archive line for the drill-down, which the writer's own comment calls "one uncomment away" -- produces a file the guard cannot distinguish from its own no-preview output, and the next walk silently replaces it. Either the writer stamps a line the guard requires, or the header stops promising. Needs scripts/mother_cat.py in context.
[[[DIVIDER]]]
- TODO (2026-09-16, the router's own drill-down; narrowed 2026-09-26): the invited edit, uncommenting the archive line beneath the preview, now leaves two uncommented paths, so the file reads as a human's and the next walk appends. The swap (comment the preview, uncomment the archive) still reads as writer-made and is replaced whole, and the header does not say so. One line the guard requires, or one sentence in the header; needs scripts/mother_cat.py in context.
[[[REPLACE]]]

3. DANGLING

- A real walk's console: "appended WALK " then "replaced this walk's own block"; the fixture has seen both, the ride has seen neither.
- The rabbit's caption names the model as the hand; the seal cannot; a wording ruling, then a reseal.
- imports/ascii_displays.py's seven adhoc.txt hits are in the trailing comment gallery, not in sealed art; a sed, not a reseal, correcting my prior turn.
- The connectors (README plus seven files), GLOSSARY.md (3), the two SKILL.md files and two_arm.py: the rename's last homes.
- The six dated adhoc.txt lines in foo_files.py and its stale epr mentions (the editor TODO,

the 2026-09-19 header TODO): a forget ride.
- The sounds: (3) whether they follow the voice consent, (4) a console line when the loop starts and the ding fires.
- The gauge's -1 branch.
- The piano box by eye: figurate player_piano, and the reseal follows what the screen shows.
- Rulings carried: the reading frame stays compile --frame lean or gets a word; PIPULATE_ADHOC_FILE, ADHOC_CHOP, the ADHOC SLOT markers and walk.py's adhoc_mutation key, each a gated car with its readers in context; page 3 of the walk naming prompt and compile beside its question, a nixops.sh truck; GLOSSARY.md's purge, a ride of its own; qamy.ai, one nginx stamp and one default.
- The 2026-09-06 NOTARY FINGERPRINT TODO stands: the Manifest still emits no wc -c line, and the \k template still says the deed's name is not in context while the footer carries it.
- The Mac: none of this has been pulled there.

4. SEED

```
# --- NEXT RIDE (seeded 2026-09-26 at the dismount): the rename's last homes, and the seal read once ---
imports/ascii_displays.py      # <-- seven adhoc.txt lines in the comment gallery, and the rabbit's caption
# connectors/README.md         # <-- the connectors' eight adhoc.txt mentions start here
# GLOSSARY.md                  # <-- three adhoc.txt lines; the purge is its own ride
# scripts/mother_cat.py        # <-- after one real walk, for its console
# --- PROBE ECHOES ---
! rg -c --hidden --glob '!.git' 'adhoc(walk)?\.txt' . | sort
! .venv/bin/python -c 'import binascii; from imports.ascii_displays import FIGURATE_REGISTRY as R, FIGURATE_LEDGER as L; [print("ART", n, "ledger=%d computed=%d" % (L[n], binascii.crc32(R[n]["render"]()[1].encode("utf-8")))) for n in ("white_rabbit", "player_piano")]' 2>&1 | tail -n 2
! rg -c 'WALK walk-.* START' "${PIPULATE_ADHOC_FILE:-context.txt}" || echo 0
```

5. CLOSING

The file that this article was written through changed its name while the article was being written, and the tool that writes that file was rewritten in the same sitting, by a model reading a payload that carried both. Nothing about that is exotic once it is on the page: two names became one, four words became three, a refusal became an append, and each step left a diff between two identical readings. The strangeness the operator set out to purge was never in the mechanism. It was in the vocabulary, and vocabulary is the cheapest thing in a repository to change and the most expensive to leave alone, because every stranger pays for it first. Operation Stick Bug is the book's long argument that an instrument earns its keep by disappearing, and this ride is the plainest form it has taken: a person types context, then prompt, then compile, and a chatbot receives exactly what those three words say. The walk, which exists to hand a newcomer that loop without the author in the room, now writes to the

same file the loop reads, and steps around a file it did not write. There is nothing left to explain about the names because the names explain themselves.
The one lesson that did not come from a plan came from a hand. The rabbit's words were changed in an editor, the seal read drift, and it read drift the same way it would have for a model. Rung one of the receipt ladder knows that something changed and never who, which is all it ever claimed; the caption that says otherwise is now on the list to fix. The four days of dark before this ride were the reminder that a record on one circuit is one record; the seal is the reminder that a record with a digest is still only a record, and the name on it is a person's, written by hand, after reading the diff.

6. NOTARIZE

Fingerprint, from the sealed bytes' own accounting: Verified Tokens 303,607; Total Bytes 1,222,094 (UTF-8); Total Chars 1,202,539; eight live command receipts in the Manifest; Codebase 181,061 tokens across fourteen entries; git-diff telemetry 385 tokens; the newest RECEIPTS entry dated 2026-09-24 (Car 1's 2026-09-26 line lands after this seal). No wc -c line exists in the Manifest to attest; that gap is the standing 2026-09-06 TODO. The deed is read off the seal footer beneath the Prompt, not guessed:

Deed: foo-fce66d3b-1567.zip (python scripts/foo_cartridge.py foo-fce66d3b-1567.zip prints the digests and exits nonzero on any tampering)

Book Analysis

Ai Editorial Take

What stands out most sharply in this piece is the realization that an AI assistant can be deployed to systematically dismantle the user's own obfuscated terminology. Typically, developers accumulate cognitive cruft—private abbreviations, pet metaphors, customized aliases—until onboarding anyone else becomes impossible. Here, Claude Fable 5.1 acts not merely as a code-generation engine, but as an editorial scalpel, methodically finding every reference across shell scripts, Lua configurations, and documentation, ensuring the developer's move toward simplicity doesn't leave lingering syntactic orphans in the dark.

Content Potential And Polish

- **Core Strengths:**
- Incredible technical transparency with real-time terminal captures showing before/after causal boundaries.
- Clear philosophical grounding contrasting ephemeral LLM inference state with permanent file-backed transcripts.
- Practical demonstration of exact-match interlocks preventing patch hallucination and silent code corruption.
- **Suggestions For Polish:**

- The compliance mapping table (NIST, ISO, SOC 2) is a hidden gem that deserves its own dedicated section earlier in the piece.
- Clarify earlier in the text why the ASCII art wax seal uses CRC checksums to detect model whitespace hallucinations.

Next Step Prompts

- Draft a dedicated chapter synthesizing the compliance crosswalk table (NIST, SOC 2, ISO 27001) with the exact git and patch mechanics used in Pipulate.
- Create a minimal tutorial explaining how newcomers can run their first three-beat walk using only context, prompt, and compile.