DESIGN OVERVIEW

Prepdeck

AN AI CO-PILOT FOR EDUCATORS

Practical support for real tasks. Stronger AI skills for what's next.

A short walkthrough of what Prepdeck does, how it's structured, and the reasoning behind its major design decisions.

DESIGNED AND BUILT BY

Nishita Grace Isaac

grace24isaac@gmail.com

24/7 INSTRUCTIONAL DESIGN

01 — PURPOSE

What Prepdeck is for

A coach, not a course — and not a tool directory either.

Most AI tools for educators fall into one of two camps: general-purpose assistants that are powerful but assume prompting expertise the user doesn't have, or narrow platforms that are easy to use but don't teach anything transferable. Prepdeck is built to sit between them — an agent that helps an educator solve a real problem this week, while making visible *why* each step worked, so the skill carries over to next month's problem without Prepdeck in the room.

1 Value before instruction

The educator brings one real task — not a hypothetical — and gets a usable result. Fluency is built as a byproduct of doing the work, not as a prerequisite to it.

2 One task at a time

A prompt aimed at a single, specific task consistently outperforms one asked to cover several at once. Prepdeck holds this constraint from the first screen, rather than letting scope creep in mid-session.

3 The educator stays the expert

Every workflow Prepdeck generates draws an explicit line between what AI can draft or organize and what requires the educator's professional judgment — about students, about their subject, about what's actually true. That line is never blurred for convenience.

02 — STRUCTURE

How the experience is organized

Three phases. Brief it, match it, build it.

LANDING → THE BRIEF → THE MATCH → THE BUILD

PHASE 1

The Brief

Role · the task, in the educator's own words · guardrails · a structured read-back

The educator describes their problem the way they'd describe it to a colleague — not the way a form wants it phrased. Prepdeck parses that into goals and guardrails, then shows its work back as editable bullets, so nothing downstream is built on a misreading.

PHASE 2

The Match

AI experience level · tools the educator can actually reach · a general recommendation

Named after the field's own principle: match the tool to the job, not the job to whichever tool is already open. This phase establishes what's actually available — a district-approved tool matters more than a theoretically stronger one the educator can't use — before any workflow is built on top of it.

PHASE 3

The Build

A human-plus-AI workflow · a prompt per AI step · a fidelity check · open troubleshooting

The workflow names, for every step, whether it belongs to the educator or to AI — and why. The generated prompt is checked not for how well it's written, but for whether it actually solves the problem stated in the Brief. If the educator runs into trouble after leaving the app, a chat carries the full session forward instead of starting cold.

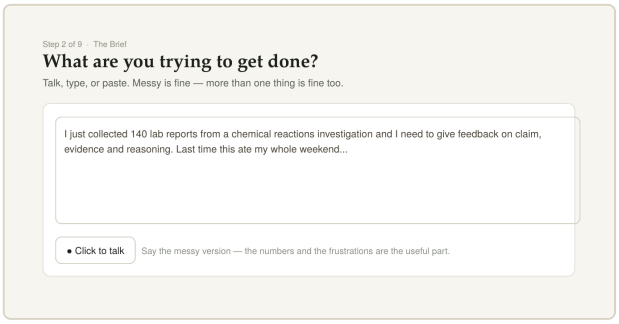
A right-hand panel carries the key facts from each completed phase into the next one, so the educator's own words — not a form summary — are what every later screen is built from.

03 — WHAT IT LOOKS LIKE

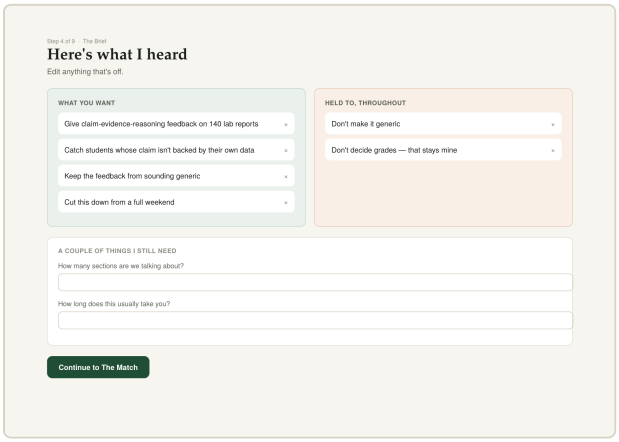
The three phases, in the browser

Rendered from the same colors, type, and copy as the working build — reconstructed here rather than captured live, since this document was prepared outside a browser session.

PHASE 1 · THE BRIEF

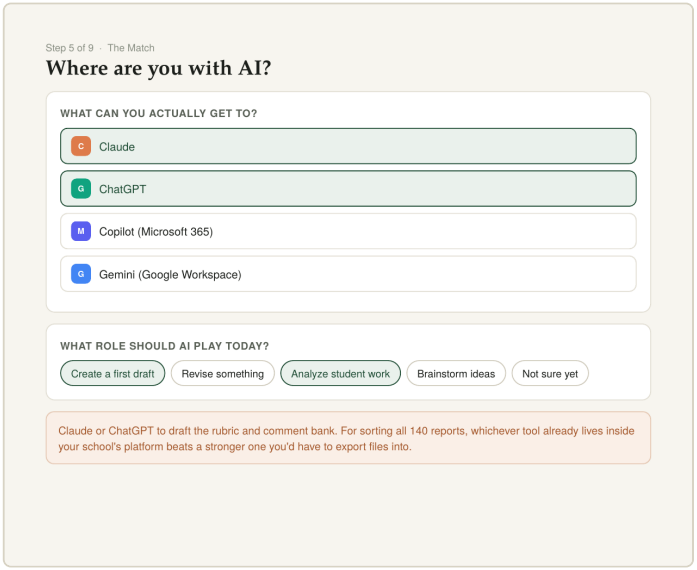


The educator's own words, unstructured — with a nudge toward the specifics (scale, prior frustration) that make a prompt useful later.



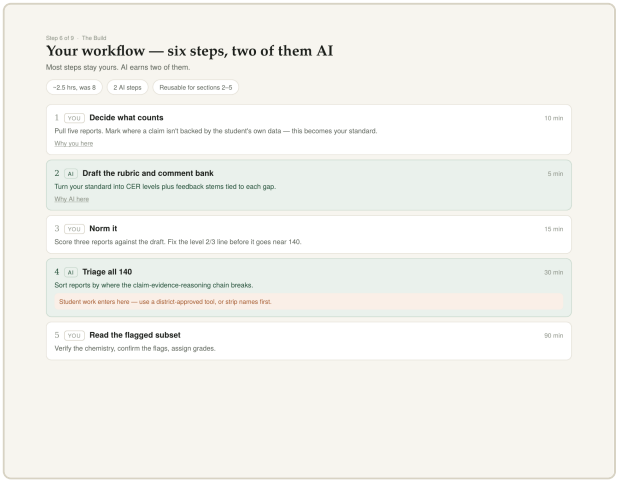
Goals and guardrails parsed back as editable bullets, visually separated by color so the two are never confused.

PHASE 2 · THE MATCH

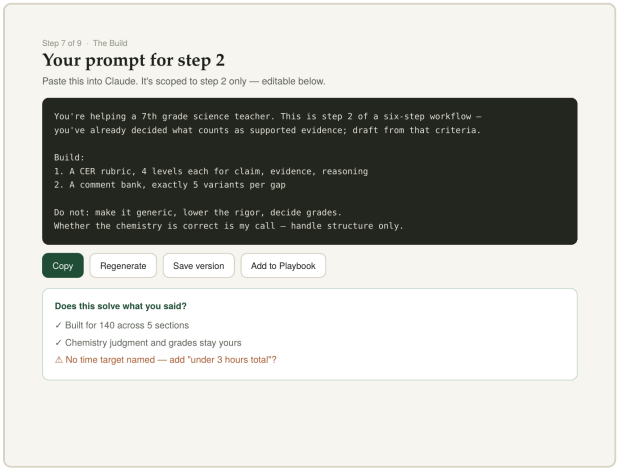


Tool access captured with recognizable logos, a free-text option for anything not listed, and the "what role should AI play" question folded in rather than asked twice.

PHASE 3 · THE BUILD



Every step marked YOU or AI, with the reasoning one click away — the moment the product's central claim has to actually hold up.



An editable prompt, checked against the goal stated in the Brief rather than against a checklist of parts.

04 — DESIGN DECISIONS

What was chosen, and why

Each of these came from a specific failure mode in an earlier version — not a stylistic preference.

Free text before structure	Early versions asked for role, task, and constraints as separate form fields before the educator ever described their actual problem. That produced technically complete but generic prompts. Prepdeck now asks for the messy version first — including scale and past frustration — because that's the information that makes a prompt specific instead of filler.
A visible "what I heard" step	Parsing free text is not guaranteed to be correct, so nothing downstream is generated silently from an unconfirmed read. Goals and guardrails are shown back as editable bullets — color-coded to keep the two apart — before the workflow is built.
Fidelity over form	An earlier scoring pass rewarded prompts for containing the right *parts* — role, task, audience — regardless of whether the result solved the educator's actual problem. Prepdeck's check now asks one question instead: does this solve what you said, not does this look complete.
Tool recommendations without a house favorite	Tools are matched to a job type — drafting, bulk sorting, visual output — not defaulted to one product. A step involving student data is weighted toward whatever tool already sits inside the educator's approved environment, even when a stronger tool exists elsewhere.
The workflow names who does what	Every step is marked as belonging to the educator or to AI, with a one-line reason. Steps needing judgment about specific students, families, or subject-matter correctness stay with the educator by design — AI is never assigned a step it shouldn't hold, regardless of how well it might perform it.
Rotating, not repeating, learning notes	Each stage carries a small bank of observations about why a particular move works — phrased as noticing, not lecturing — drawn at random so the same explanation doesn't repeat verbatim across sessions or wear thin on return visits.

05 — INTENDED EXPERIENCE

How it should feel to use

Progressing toward the task, not filling out a form.

An educator arrives with one real thing due this week — a set of lab reports, a parent letter, a data review. They describe it in their own words, confirm what Prepdeck understood, say what they have access to, and receive a workflow built specifically for their situation rather than a generic template with their name inserted.

At every stage, a short note makes visible *why* the current step matters — not as a lesson to sit through, but as a comment made in passing while the real work continues. Nothing about the interaction should feel like a detour from the task; the fluency is what's left over once the task is done.

THE TEST THIS IS HELD TO

If an educator finishes a session and can explain, unprompted, why one step used AI and the next one didn't — the design has done its job. A good output alone isn't enough.

06 — NOTES FROM BUILDING IT

What actually went into this

A few things worth naming honestly, rather than smoothing over.

> **The hardest bug wasn't a bug in the code — it was a bug in the safety net.** An early version quietly fell back to placeholder content whenever generation failed, so a user typing anything at all would eventually see someone else's already-completed example instead of their own. It took real interrogation of the state-management logic to find, because everything downstream of the fallback was still functioning "correctly" — it was just correctly displaying the wrong thing. The fix wasn't a patch; it was deciding that a visible error is always preferable to an invisible substitution, and rebuilding the failure path around that rule.

> **I rewrote the intake flow from scratch once, on purpose.** The first version worked — every field validated, every step was reachable — and it still felt like paperwork. The problem wasn't a missing feature, it was sequencing: asking about AI before asking about the actual problem. Recognizing that a fully functional flow could still be the wrong flow, and being willing to restructure instead of polish, was the harder call to make than any individual screen.

> **Deciding what AI shouldn't be allowed to recommend was as much work as deciding what it should.** A tool-matching feature that always suggests the same product regardless of the job isn't a recommendation engine, it's a bias with a UI. Making the system name a tool's actual weakness, and sometimes recommend no AI tool at all, meant writing that constraint explicitly rather than trusting it to emerge on its own.

> **Testing against more than one persona caught things a single test case never would have.** A workflow that looked reasonable for a science teacher's lab reports produced a genuinely different — and genuinely necessary — shape when tested against a principal's parent newsletter. Holding both in mind at once, instead of generalizing too early from whichever one I built first, is what kept the tool-matching logic honest.

> **I had to get comfortable shipping something I knew was still wrong in places.** Environment constraints — a sandboxed preview that blocks microphone access, for instance — meant some features could be built correctly and still fail to demonstrate correctly in every context. Documenting that distinction plainly, instead of either hiding it or holding up the whole project for it, was its own kind of judgment call.

This document accompanies the working prototype. Screens shown here reproduce its actual design system and copy; prompts and workflows shown elsewhere in this application were produced by the live system directly.