PUBLISHED: SEPTEMBER 9TH · BY SHREYAS KALYANPAD

An agent-built interface = a screen a coding agent produced from a prompt, with none of your design system attached.

These screens are not bad. They are fine. They are also identical to every other screen, and a reader clocks that in about two seconds.

Most people call this a taste problem. It is a context problem. Every tell below is a gap you left, filled in with the average of the internet.

> *The model is not guessing wildly. It is guessing the average. The average product screen of the last decade is a sidebar and a grid of cards.*

The six tells

For each: what you see, why it happens, what to feed the agent instead.

Every one of these was meant to be a different product.

1. Every screen is the same dashboard

What you see: sidebar, top bar, stat cards, repeat. A settings page and a triage queue from one layout.

Why: a dashboard is the safe general answer, so it wins when nothing else is specified.

Feed it: the job of the screen. "List and detail, for triage" beats "a page for orders".

2. Every card weighs the same

What you see: the number that runs the business in the same box as a vanity stat.

Why: nothing told it which one matters, and an even grid is cheapest.

Feed it: one hero element per screen, named. Everything else is secondary by default.

3. One spacing value, used everywhere

What you see: even gaps, no grouping, nothing for the eye to rest on.

Why: this is the important one. If your system is only readable on request, the agent asks for the button and invents the spacing.

Feed it: spacing, colour and type as always-on rules. Indeed's design system team keeps foundations always-on for this reason.

4. Stock components, untouched

What you see: the default radius, the default shadow, the default focus ring. A stranger can name the kit.

Why: the fastest working layout is an off-the-shelf library, tokens untouched.

Feed it: three or four primitives with an opinion. You do not need the whole system on day one.

5. No empty, loading or error states

What you see: a blank first run, a raw error string, a spinner.

Why: generation optimises for renders and functions. These states are off the happy path.

Feed it: the three states as part of the request, every time. This is the strongest signal of human care left.

6. Copy any product could have written

What you see: "Welcome back", "Manage your account", "Something went wrong".

Why: correct language is easy. Specific language needs to know what things are called.

Feed it: your real nouns. Rewrite the primary buttons, empty states and top errors first.

Why a longer prompt will not save you

The usual answer is to write more rules into a guidelines file. Figma's Laura Fehre puts it flatly: in nearly all cases the prompt wins over the guidelines. The file is not a law.

The measured answer is structure. Diana Wolosin ran 1,056 prompts against Indeed's real design system across eight setups. Piping the human documentation straight in was about five times more expensive, and less accurate, than the same system written as structured JSON. Against a Markdown and JSON hybrid, the JSON used 80% fewer tokens at equal or better accuracy.

So the lever is not prose. It is machine-readable specifics, always on.

The swap

THE TELL	WHAT IS ACTUALLY MISSING
Same dashboard everywhere	The job of each screen
Flat card grid	A named hero element
Even spacing	Foundations as always-on rules
Stock components	Three primitives with an opinion
Missing edge states	Empty, loading and error in the request
Generic copy	Your product's real nouns

A ten minute audit

Open the last thing your agent built.

1. Squint at it. Can you still tell what matters most?
2. Put your main screens side by side. Are they the same layout with different data?
3. Could a stranger name the component library?
4. Open the empty state. Is there one?
5. Read the buttons aloud. Do they name real things in your product?

Every no is a gap the model filled for you. Write it down as a rule, not a line in a prompt, and the next screen starts further along.

Checked against

Read on 9 September 2026.

- Into Design Systems: Diana Wolosin's 1,056-prompt benchmark on Indeed's design system, and Laura Fehre on guidelines losing to the prompt
- SaaSUI.Design: the seven signs a SaaS interface reads as AI-generated, with fixes
- Figma's 2026 AI report: 8,403 responses on how AI changed design work
- Figma: why design systems are the context AI agents are missing
- Kevin Coyle, zeroheight: why long prompts lost to structured component specs, and what MCP changed