

How to make your design system agent-ready.

8 phases and 20 rules that make AI build with your design system, not around it. Each rule says what to do and who does it.



Ravi Surana



THE FACT

71% of design system teams use AI for code. Only 12% give it their docs.

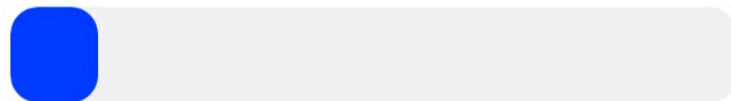
Most teams leave their AI to guess.
Here is how to close that gap.

Use AI to write code



71%

Give AI their design system docs



12%

Share of design system teams. zeroheight Design Systems Report 2026, 147 respondents.



Ravi Surana



The AI design system playbook.

0 · Know what you have

1 · Tokens

2 · Components

3 · One master copy and docs

4 · Findable

5 · Page templates

6 · Test it

7 · Keep it fresh

8 phases and 20 rules, one rule per slide.
Each says what to do, why, and who does it.



0

Know what you have.

This phase has 2 rules.

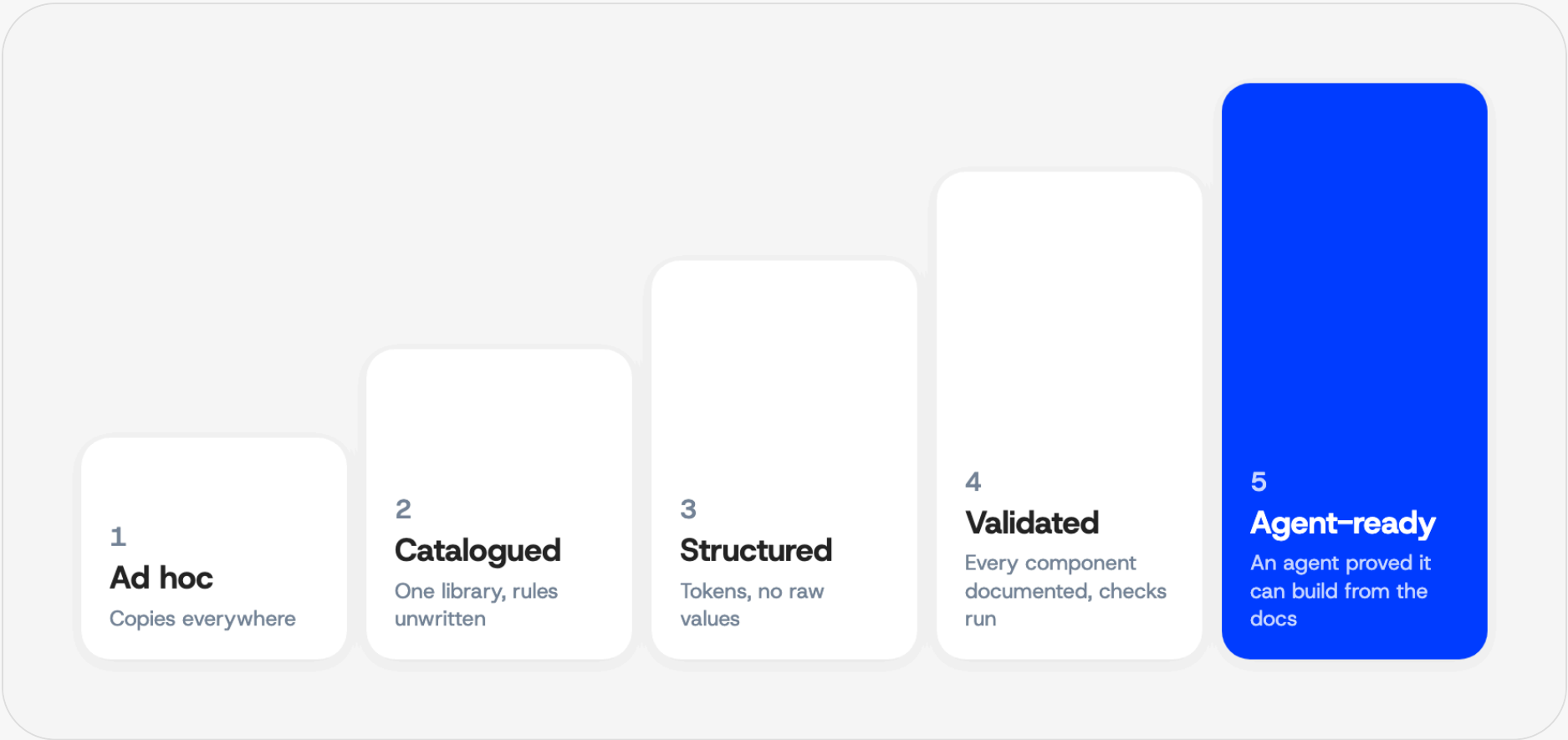


Ravi Surana

Score your system from 1 to 5.

Many teams are at 2: one library, with the rules in people's heads. Pick one real screen as your answer key. Ask a fresh agent to build it, and save what it gets wrong.

Why: you can't fix a system until you know how far along it is.



Owner: PM + Design



Count what you have. Pick one person to own it.

Count every token, component, doc and hand-typed colour or size. Then pick one person whose job is to keep the docs up to date.

Why: nobody can fix what nobody has counted,
and a doc with no owner goes stale.

WHAT YOU HAVE · EXAMPLE COUNT

Tokens	142
Components	38
Components with a doc	12
Hand-typed values	211

DOCS OWNER



One named person

Keeps every doc true.
Not a team, not
"everyone".

Owner: PM + Design + Eng



1

Fix your tokens.

This phase has 3 rules.

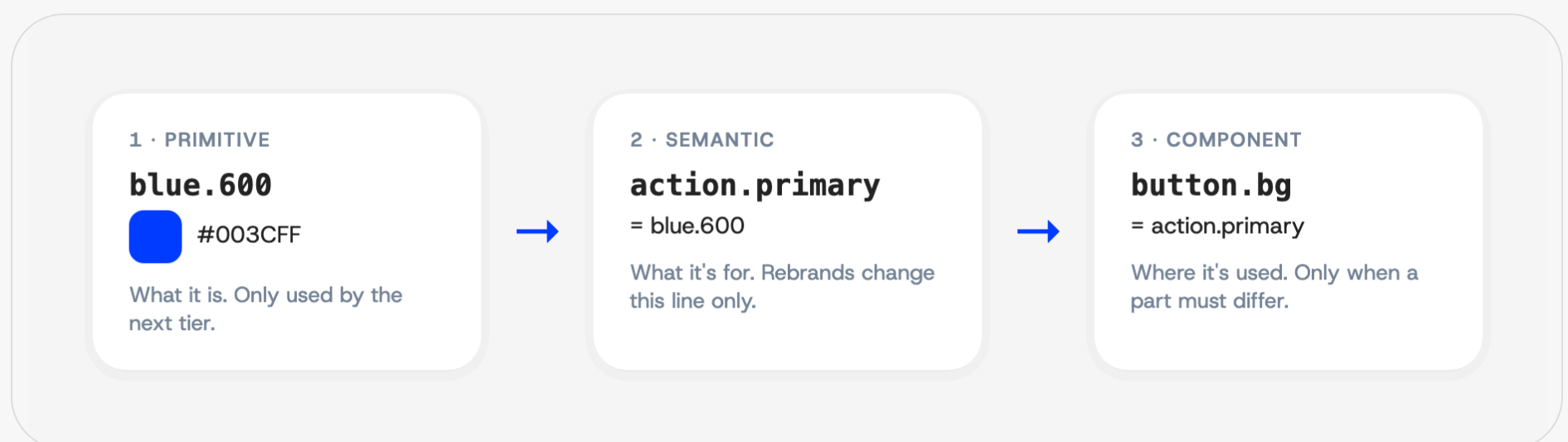


Ravi Surana

Name tokens by what they do, not how they look.

Tokens are named colours, sizes and spacing.
Call it `action.primary`, not `blue-600`.

Why: `blue-600` says how a colour looks. `action.primary` says what it is for.



Owner: Design + Eng



Use the standard token format.

Use DTCG, the format from the W3C Design Tokens Community Group. It has been stable since October 2025, and it handles light and dark mode too.

Why: a standard format lets different tools read the same file without custom code.

tokens/color.tokens.json

W3C DTCG

```
"action": {
  "primary": {
    "$type": "color",
    "$value": "{color.blue.600}",
    "$description": "Main action on a screen"
  }
}
```

ONE FILE, READ BY

Design tools

Code, on every platform

AI agents

Light and dark mode live in the same format.

Owner: Eng



Ravi Surana

Make hand-typed values fail the build.

A hand-typed value is a hex colour or a one-off size like `p-[13px]`. Add a check that rejects them, and let agents run the same check.

Why: a hand-typed value can be off by one pixel and nobody notices.

✗ What agents write by default

```
background: #1D4ED8;  
padding: 14px;  
border-radius: 6px;
```

Close enough. Wrong everywhere.

✓ What your lint rule allows

```
background: var(--action-primary);  
padding: var(--space-inset-md);  
border-radius: var(--radius-control);
```

One source. Changes everywhere at once.

Owner: Eng



2

Fix your components.

This phase has 4 rules.



Ravi Surana

Clean up your design file.

Give every layer a real name, not Frame 1268. Use variables for colour, spacing and type, and let frames resize on their own (auto layout in Figma).

Why: agents read your design file through the tool's MCP server. A layer called Frame 1268 tells them nothing.

LAYERS · BEFORE

Frame 1268

- Rectangle 4
- Group 12
 - Text
 - Text

Frame 1271

→

LAYERS · AFTER

Card / Product

auto layout

Image

radius.photo

Title

text.title

Price

text.body

Button / Primary

action.primary

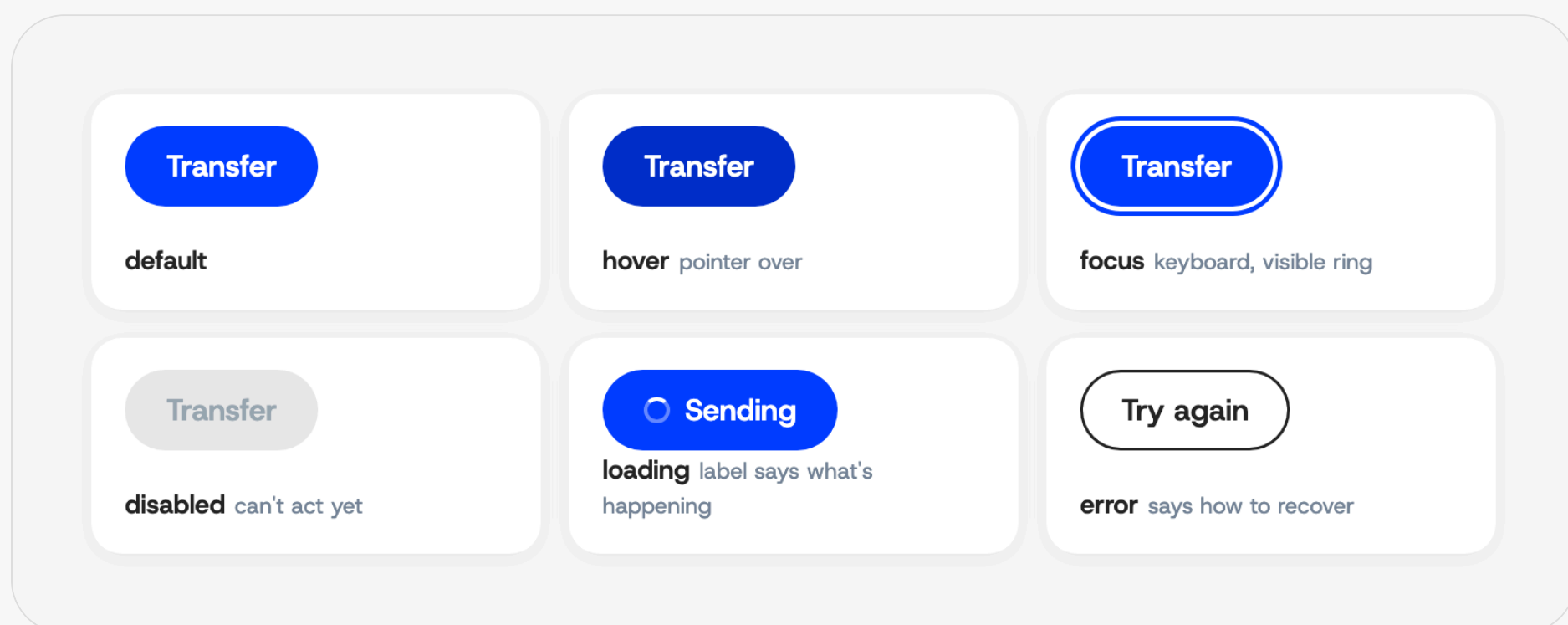
Owner: Design



List every state and option. Allow nothing else.

A state is how a part looks when hovered, disabled or loading. List every state, size and style. Unlisted means the agent invents it.

Why: if a state is not written down, an agent has to guess it.



Owner: Design + Eng



Write down the limits.

A button group holds 2 to 4 buttons, never nested.
Say it, or the agent can't know.

Why: an agent can't see limits that only live in your head.

BUTTONGROUP · HOLDS 2 TO 4 BUTTONS, NEVER NESTED

✓

3 buttons

✗

6 buttons

✗

Nested group

Owner: Design



Map every design part to its code twin.

Match each component in your design file to its code component. A simple table works with any tool. Figma also has Code Connect on Organization and Enterprise plans.

Why: a design part with no code twin gets rebuilt from scratch, and the copy comes out different.

IN THE DESIGN FILE		IN CODE
Button / Primary / Large	↔	<Button intent="primary" size="lg">
Input / With label	↔	<TextField label="...">
Card / Product	↔	<ProductCard>

A plain table works with any design tool.

Owner: Design + Eng



3

Keep one master copy and docs.

This phase has 5 rules.



Ravi Surana

Keep one master copy, as plain files.

Tokens, one doc per component and a component list, all in a design-system/ folder. Code builds from it. Figma syncs from it through a plugin like Tokens Studio, or its API on Enterprise.

Why: when the same rule lives in the design file, the code and a wiki, the copies drift apart.

THE MASTER COPY

- ▼ design-system/
 - DESIGN.md
 - ▼ components/
 - button.md
 - card.md
 - ... one doc each
 - tokens/
 - manifest.json



Design tool

syncs from it



Code

builds from it



AI agents

read it before they build

Owner: Design + Eng



One doc per component, one screen long.

Cover when to use it, when not, states, parts, tokens and accessibility.
Keep long reasoning separate.

Why: long docs get skipped. A short doc gets read by people and by agents.

button.md also called: CTA, action button

RULES

- MUST** One primary button per card.
- MUST** Text takes its fill's paired colour.
- SHOULD** Label starts with a verb.
- SHOULD** Links navigate. Buttons act.

PARTS

Button

- Label
- Icon (optional, max 1)

default hover focus disabled loading

Owner: Design



Show one right example and one wrong one.

Give each component a short code example that works when pasted. Next to it, show a common mistake marked as wrong.

Why: agents copy the examples they see. A marked wrong example shows the limit faster than a paragraph.

button.md EXAMPLES

✓ Right

```
<Button intent="primary">Save changes</Button>
```

✗ Wrong: styled by hand, so it skips the tokens

```
<button style="background:#1D4ED8">Save</button>
```

Owner: Design + Eng



Give every rule a strength.

MUST is never broken. Add a check that fails the build where you can. SHOULD can bend. An untagged rule looks optional to an agent.

Why: an agent can't tell a hard rule from a suggestion unless you say which is which.

THE RULE, AS WRITTEN	WHAT THE AGENT DOES
MUST One primary button per card.	Stops. The build fails if broken.
SHOULD Labels start with a verb.	Follows it. Can bend with a reason.
NO TAG Keep cards simple.	Treats it as optional.

Owner: Design



Write what not to do, and why.

A clear “never” beats a vague “prefer”. Write one for every component and every token. Name what to use instead, and which rule wins when two clash.

Why: a vague “prefer” leaves room to argue.

A “never” with a reason does not.

`button.md`

✕ **Never put two primary buttons in one card.**

Use instead: one primary, one secondary.

Why: the eye needs one next step.

`action.primary`

✕ **Never use it for text links.**

Use instead: `text.link`.

When rules clash: contrast beats brand colour.

Owner: Design + Content



4

Make it findable.

This phase has 2 rules.



Ravi Surana

Keep a list of every component and icon.

One file lists every component: each option with a short description, its states and tokens. List every icon by name too. CI rebuilds it on every deploy.

Why: an agent can't use a part it doesn't know exists, so it invents one.

manifest.json rebuilt by CI on every deploy

```
{ "component": "Button",  
  "props": { "intent": ["primary", "secondary"] },  
  "about": { "intent": "primary = the one main action" },  
  "states": ["default", "hover", "focus", "disabled", "loading"],  
  "tokens": ["button.primary.bg"], "doc": "components/button.md" }  
{ "icon": "arrow-right", "keywords": ["next", "forward"],  
  "import": "@ds/icons/ArrowRight" }
```

Owner: Eng



Point agents to your docs.

Write a short AGENTS.md, under 200 lines, that links to your docs. Cursor, Codex and Claude Code read it on their own. If you also have a CLAUDE.md, import AGENTS.md from it.

Why: agents read this file first. Long files get followed less closely, so link to your docs instead of pasting them in.

CLAUDE.md

@AGENTS.md

Only needed if you keep a CLAUDE.md. Without one, Claude Code reads AGENTS.md on its own.



AGENTS.md

UNDER 200 LINES

```
## Design system
Read design-system/DESIGN.md first.
Use only parts in manifest.json.
Use only tokens. Never raw values.
Read a part's doc before using it.
Check your work: npm run check:tokens
```

Owner: Eng



Ravi Surana



5

Add page templates.

This phase has 1 rule.



Ravi Surana

Give agents whole screens to start from.

Add a few ready page layouts, like a settings page or a list page. Build them only from your own parts. List each one in manifest.json so agents can find it.

Why: agents build whole screens, not single buttons.
Without a starting layout, they invent one.

SETTINGS PAGE · TEMPLATE

ONLY YOUR PARTS

<SideNav>

Settings

<PageHeader>

<TextField>

<TextField>

<ButtonGroup>

Owner: Design + Eng



6

Test it.

This phase has 2 rules.



Ravi Surana

Run the agent test again.

Ask a fresh agent to build the same real screen from your docs only. Compare it with your answer key and with your first run.

Why: docs can look complete and still fail the moment someone follows them.

SAME SCREEN · EXAMPLE MISSES	FIRST RUN	NOW
Made-up parts	4	0
Hand-typed values	11	1
Missing states	5	1
Broken MUST rules	3	0

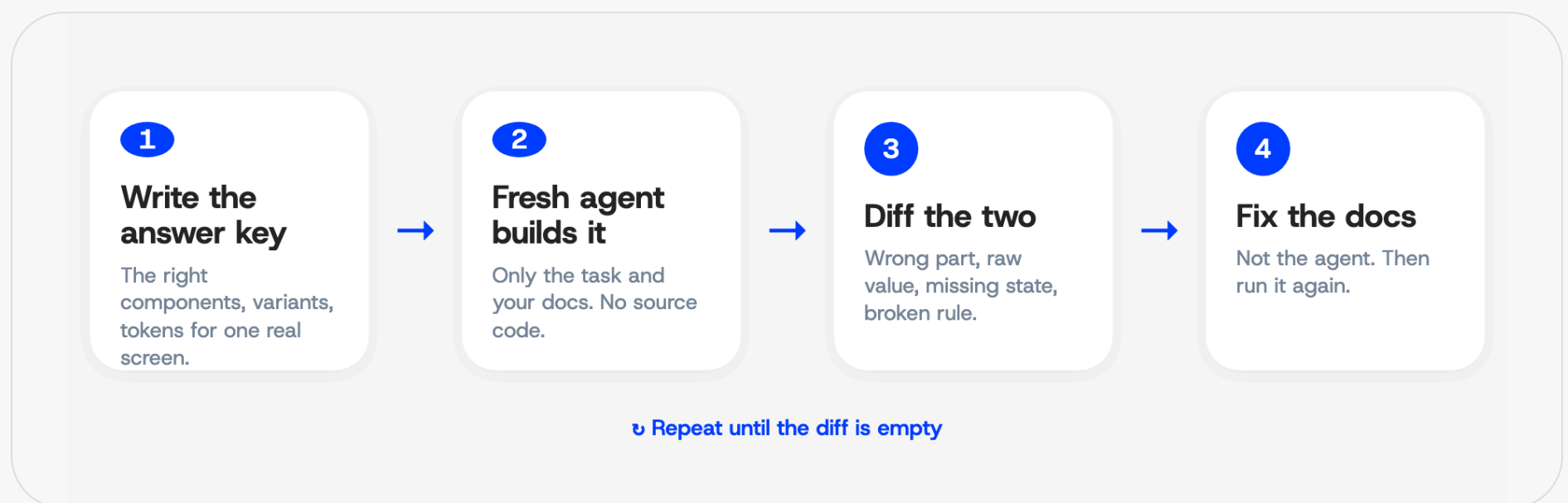
Owner: Eng + Design



Fix the docs, not the agent.

Score each run: parts used correctly, hand-typed values, missing states. Every miss is a vague rule in your docs. Fix the doc, then re-run.

Why: when an agent gets it wrong, a vague sentence in your docs is usually the cause.



Owner: Eng + Design





Keep it fresh.

This phase has 1 rule.



Ravi Surana

Ship docs with the code.

A component change without its doc change fails CI. Also check for drift (design, code and docs disagreeing) on every change.

Why: docs that change on a different schedule than the code go out of date.

Pull request · Button: add “loading” state

✓ Tokens valid (DTCG)	passed
✓ No hand-typed values	passed
✗ button.md not updated	blocked
✗ Design file and code disagree on states	drift found

Owner: Eng



Ask an agent to add a popup.

Each file answers one question, in this order. The agent never has to guess.

Why: if a file is missing, that step turns back into a guess.

Request: add a confirm popup to Settings.

- | | | |
|---|---------------|---|
| 1 | AGENTS.md | Says: read DESIGN.md first. |
| 2 | DESIGN.md | Gives the rules and where everything lives. |
| 3 | manifest.json | Confirms a Dialog exists and lists its options. |
| 4 | dialog.md | Gives its states, limits and what never to do. |
| 5 | tokens/ | Supplies colour, radius and spacing. |
| 6 | the check | npm run check:tokens passes. Done. |





ATLASSIAN · MAY 2026

Atlassian set up its system for AI agents.

+52%

AI accuracy

34%

faster on design-system tasks

-16%

AI tokens used

-26%

AI tool calls

Atlassian reported these numbers itself.
Source in comments.



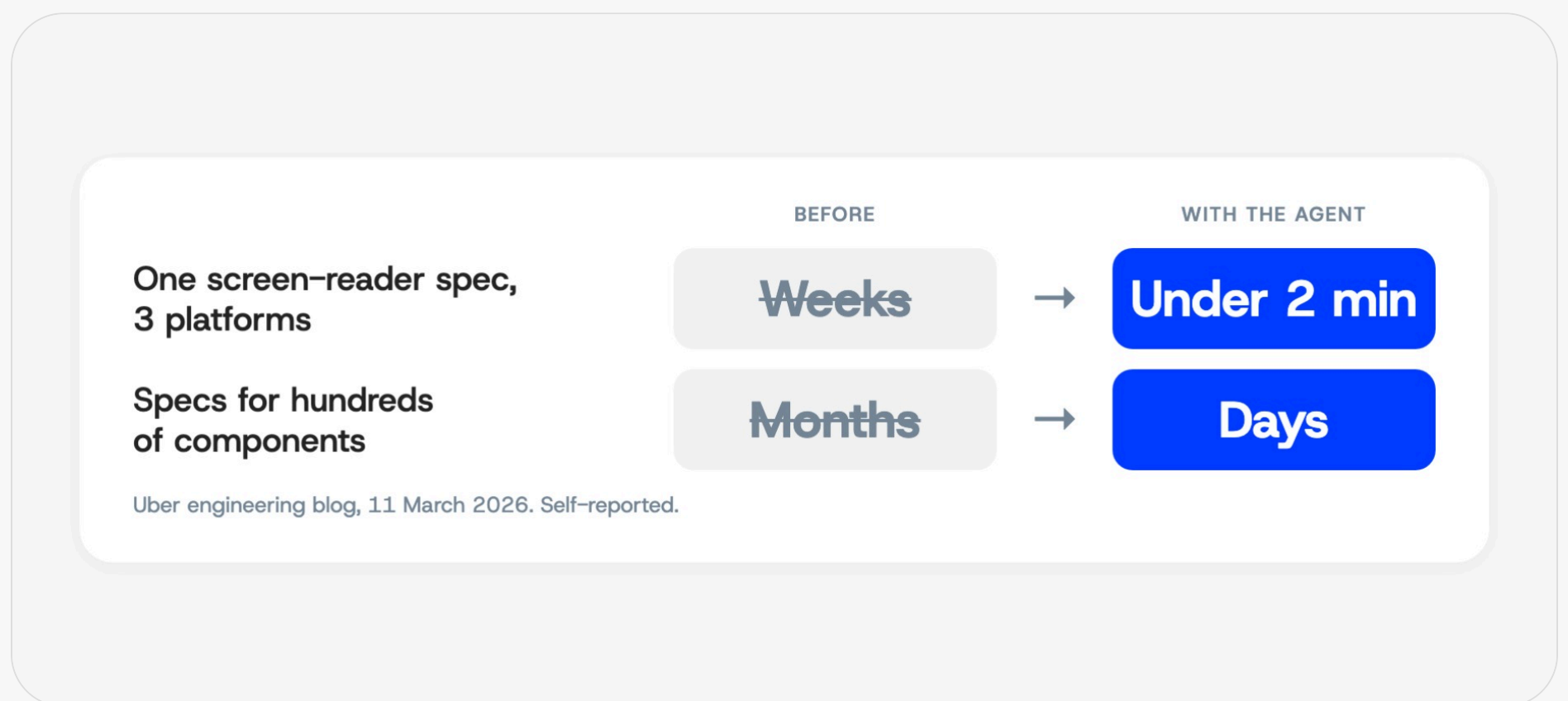
Ravi Surana



UBER · MARCH 2026

An agent writes specs from the design system.

Uber did it because stale specs meant engineering “builds from assumptions instead of definitions.”



Ravi Surana

coinbase

COINBASE · 2026

Coinbase linked each component to its code.

–22.3% **–11.5%**

time to implement a design

AI tokens used

Figma published these numbers. Source in comments.



Ravi Surana

35



WHAT THAT MEANS FOR YOU

Five wins from one well-written system.

AI costs less

Builds go faster

AI output stays consistent

Fewer handoff errors

Designers and engineers stay aligned

Every phase serves at least one.



Ravi Surana

Write it down, or the AI will guess.

Message me if you need help making your
design system AI agent-ready.



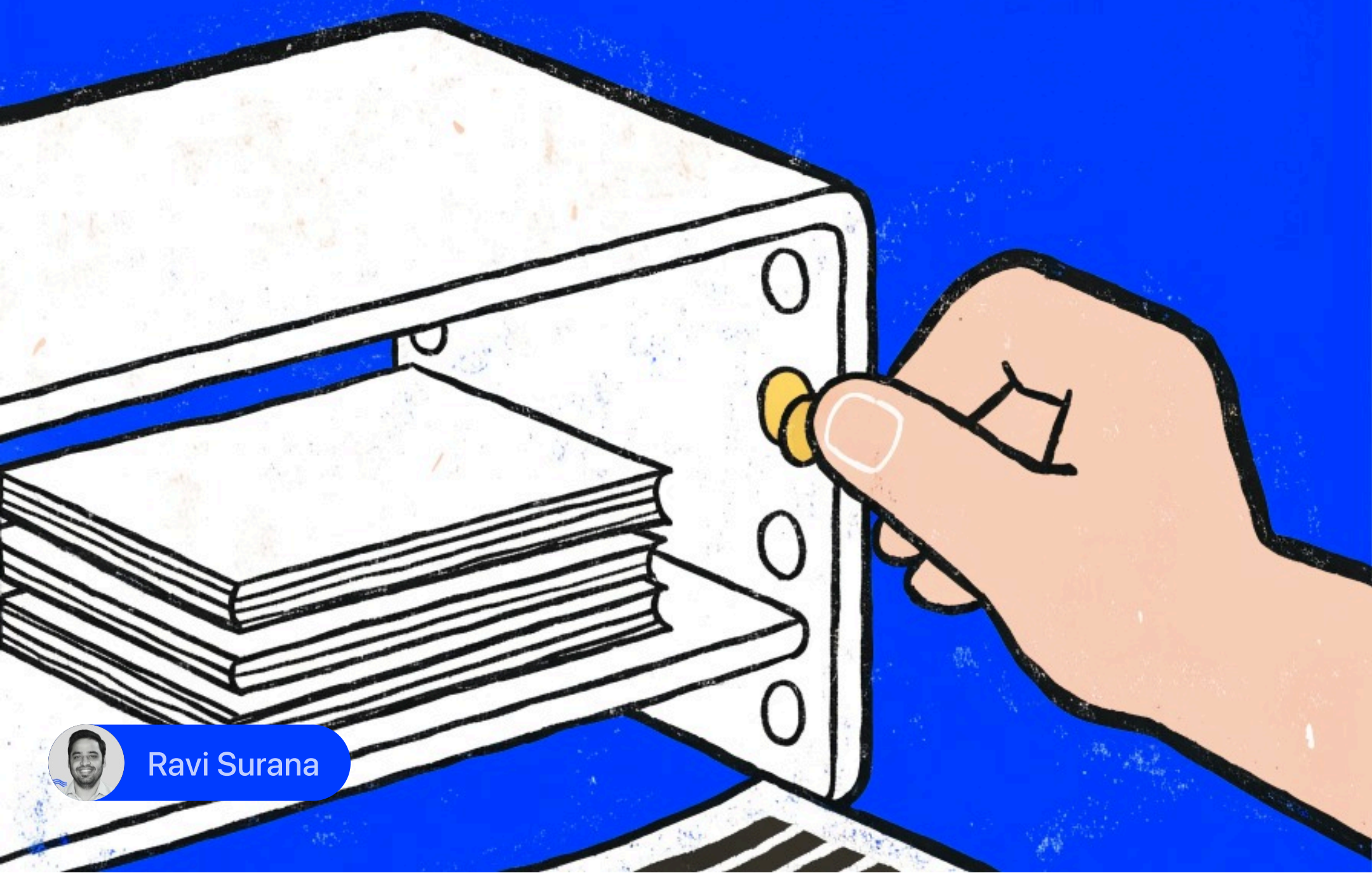
Like



Share



Download



Ravi Surana