AI DESIGNER SETUP

Great design = *a solid setup.*

And it comes down to **5 things.** How to write each one with your own design judgment.

(01)
CLAUDE.md
The project brain

(02)
DESIGN.md
Visual rules

(03)
Rules
Hard limits

(04)
Skills
Repeated tasks

(05)
Memory
Learned habits

READ FIRST

How to use *this guide.*

Great design doesn't come from the prompt. It comes from what the AI reads before it: your setup.

This is not a ready-made kit. Each section gives you a structure and the questions you answer with your own criteria. Copying the example gives you generic results. Filling it with the way you work is what makes the difference.

(00) / TWO PLACES

Global vs. *project setup.*

Global holds how you work and applies to every project. Project holds what belongs to that product only.

GLOBAL

`~/.claude/`

Your method. Every project.

```
└ CLAUDE.md
└ rules/
└ skills/
```

PROJECT

`my-app/`

That product. Ships with the repo.

```
└ CLAUDE.md
└ DESIGN.md
└ .claude/rules/
└ .claude/skills/
```

Ask yourself. Does this apply to any product I work on? If yes, global. If it depends on that product, project.

(01) / CLAUDE.MD

The project *brain.*

Claude reads this file before any request you make, so it has to be short: just the essentials, pointing to the rest with `@`.

Where it lives: `./CLAUDE.md` in the project. For preferences across every project, `~/.claude/CLAUDE.md`.

STRUCTURE TO FILL IN

```
# Project
[What the product is, in one line]
[stack]

# Read first
@DESIGN.md

# Always
- [a rule for every task]
- [a rule for every task]

# Never
- [the mistake you've corrected
  more than once]
```

THE QUESTION EACH SECTION ANSWERS

(01) **Project:** if someone new joined today, what would they need to know in 10 seconds?

(02) **Read first:** which files does the AI need so it doesn't guess?

(03) **Always:** what do you ask for in almost every prompt? Move it out of the prompt and into here.

(04) **Never:** which correction have you repeated? Write it here once.

Warning sign. If the file is longer than one screen, move the detail into DESIGN.md, a rule or a skill.

(02) / DESIGN.MD

Where AI stops *guessing.*

It holds three different things, and knowing the difference is the point.

Tokens
The values that exist: colors, type, spacing, radius.

Guidelines
How components are used, and when not to use them.

Criteria
The decisions only a designer makes. Nobody can copy this part from you.

Where it lives: in the project root, next to CLAUDE.md. Claude Code doesn't load it on its own, so CLAUDE.md has to point to it with `@DESIGN.md`.

ANNOTATED EXAMPLE

```
## Colors            ← tokens
--bg: #FAF9F6
--accent: #FF5A1F

## Type              ← tokens
Archivo · Bold 64 / Regular 17

## Components        ← guidelines
Button: primary, ghost
Primary: one per screen

## Criteria          ← your judgment
Don't create new variants
without asking
When in doubt, less:
remove before adding
```

QUESTIONS TO FILL IT IN

(01) Which values does the AI keep inventing? Those become tokens first.

(02) Which component gets misused most? Write when to use it and when not to.

(03) What would you say in a design review that isn't written anywhere? That is your criteria section.

(03) / RULES

One rule *per topic.*

DESIGN.md says what exists. Rules say what never to do, and only where it matters. Instead of one giant file, each topic gets its own.

Where they live: `.claude/rules/` in the project, one `.md` per topic. For rules you want everywhere, `~/.claude/rules/`.

STRUCTURE TO FILL IN

```
# .claude/rules/ui.md
---
paths:
  - "src/components/**"
---
- Never [what breaks
  consistency]
- Always [what keeps
  it consistent]
```

WHY PATHS MATTER

The `paths` block at the top makes the rule apply only to those folders, so it doesn't weigh on unrelated work. Your UI rules stay out of the way when you're fixing an API.

HOW TO WRITE A RULE THE AI FOLLOWS

Specific beats general.

"Spacing in multiples of 4 only" works. "Keep spacing consistent" doesn't.

One rule per line.

Short lines are easier to follow and easier to change.

After the second correction.

A one-off fix stays in the chat. A repeated one becomes a rule.

Start here. Color only from tokens, spacing on the scale, every interactive element has a focus state, no inline styles.

(04) / SKILLS

Repeated task?
Make it a skill.

Each skill is a folder with its own `SKILL.md`. Claude only loads the full content when the task calls for it, so you can have many without weighing down the context.

```
~/.claude/skills/      GLOBAL
  audit-screen/
    SKILL.md
  create-component/
    SKILL.md

my-app/.claude/skills/ PROJECT
  client-patterns/
    SKILL.md
```

- **Global:** your method, available in every project, only on your machine.
- **Project:** what only makes sense for that product. It ships with the repo, so your team gets it too.
- **The trick:** a global skill can read the DESIGN.md of whatever project you're in. Write "create component" once and it follows each project's tokens.

ANATOMY OF A SKILL.MD

```
---
name: audit-screen
description: Use when reviewing
  a screen against the design system.
---

1. Read DESIGN.md
2. [first thing you always check]
3. [second thing you always check]
4. Report what breaks the system,
   by severity
```

The `description` decides when Claude uses the skill, so write it as "use when...".

FIND YOUR FIRST SKILL

(01) What do you explain to the AI the same way every week?

(02) Which review do you always do in the same order?

(03) If you had to teach a junior designer one process, which would it be? That's the skill.

STAY SAFE

Build your skill.
Don't grab it ready-made.

A ready-made skill is a stranger's instructions and code running with your permissions. Skills inherit the agent's full access: the shell, your files, credentials in config files, the ability to send messages, and persistent memory.

36.8%
of 3,984 audited skills had a security issue
SNYK · FEB 2026

13.4%
had a critical one: malware, prompt injection or exposed secrets
SNYK · FEB 2026

400+
fake crypto skills shipped info-stealing malware to Claude Code users in one week
OPENSOURCEMALWARE · JAN 2026

1,184
malicious skills counted on the ClawHub marketplace
ANTIY VIA HARD2BIT

90%+
of the time, malicious skills got past marketplace scanners in testing
CLOUD SECURITY ALLIANCE · JUL 2026

10x
jump in new skills published per day, in a few weeks
SNYK · FEB 2026

Keep in mind. A "verified" badge on a marketplace is not a safety guarantee. Full source links are on the last page.

STAY SAFE

How they hide, *and why it matters.*

Setup steps

Instructions that tell the agent to download and run software from outside.

Encoded commands

Text like base64 that sends your credentials to someone else's server.

Plain-text instructions

Words that hijack the agent. No code needed, so code scanners miss them.

Inside a repo you download

Claude Code loads a project's `.claude/skills`, so opening an unknown repo can bring its skills along.

SECURITY IS ONLY HALF OF IT

Someone else's criteria.

It reflects how another person works, so your output looks like everyone else's.

You can't fix it.

If you don't understand it, you won't know why it fails.

Writing it is the learning.

Turning your process into steps is what makes you better at it.

IF YOU STILL USE ONE FROM SOMEONE ELSE

- [ ] Read the whole SKILL.md and every script in the folder
- [ ] Look for red flags: downloads, `curl | bash`, encoded text, links to outside instructions, requests for keys or passwords
- [ ] Prefer official or well-known sources
- [ ] Use it as a reference, then rewrite it in your own words with your criteria

(05) / MEMORY

What it learns *from you.*

It's the only part you don't write: every correction becomes a note, and next session it already knows, so you don't have to repeat yourself.

Where it lives: automatically, one per project, in `~/.claude/projects/<project>/memory/MEMORY.md`. Only the first 200 lines load each session.

WHAT IT LOOKS LIKE

```
# MEMORY.md
- Buttons use radius 8, not 12
- Icons live in /assets/icons
- Prefer semantic tokens
```

REVIEW CHECKLIST

- [ ] Delete notes that are no longer true
- [ ] Move notes that apply to every task into CLAUDE.md
- [ ] Move notes about visual decisions into DESIGN.md
- [ ] Turn repeated processes into a skill

Review it. A wrong note becomes a wrong habit. Run `/memory` during a session to open and edit it.

NEXT STEP

Start *small.*

One good file today beats five empty ones.

- [ ] Write your CLAUDE.md with the four sections
- [ ] Create DESIGN.md with the tokens the AI keeps inventing
- [ ] Turn your most repeated correction into a rule
- [ ] Turn your most repeated process into a skill
- [ ] Open `/memory` in a week and review it

Want to see *mine?*

My e-book *AI-Powered Product Design* goes deeper: my own setup, real product cases and the full stack. A refined edition is on the way.

Follow @gustavodgirard

SOURCES

- How Claude remembers your project, Claude Code docs
- ToxicSkills: malicious AI agent skills, Snyk
- SkillCloak: malicious agent skills evade marketplace scanners, Cloud Security Alliance
- Clawdbot skills ganked your crypto, OpenSourceMalware
- MoltBot skills exploited to distribute 400+ malware packages, Security Affairs
- Malicious AI agent skills: how to vet before installing, Hard2bit