AI-AUGMENTED PRODUCT MANAGEMENT · CASE STUDY

Rapid Prototyping: How I Turn Ideas Into Working Tools with AI

From a rough problem statement to something a stakeholder can click, in an afternoon — using a version-by-version build process, not one-shot generation.

Yash Mahadik
Product Manager & Founder, AGAR

Mumbai, India
July 2026

CORE PRACTICE RAPID PROTOTYPING

Prototype First, Deck Second

My default response to a new idea, feature, or concept isn't a slide — it's a working version someone can click. A stakeholder or investor engaging with a real interface makes a faster, better-informed call than one reading a description of it. I've built this into a repeatable process rather than a one-off trick, and I use different tools depending on the surface I'm prototyping.

6
TOOLS IN ROTATION, MATCHED TO THE SURFACE

5–7
VERSIONS PER PROTOTYPE, LOGIC BEFORE POLISH

1
AFTERNOON — TYPICAL TIME TO A CLICKABLE V1

The Toolkit, Matched to the Surface

Lovable and **v0 (Vercel)** for fast, opinionated UI-first builds — landing pages, dashboards, product surfaces. **Firebase Studio** and **Google AI Studio** when the prototype needs real backend or model behavior, not just a mock. **Cursor** and **Google Antigravity** when I need to get closer to the actual codebase. **Claude** for anything conversational, logic-heavy, or where I want to reason through the build in plain language before touching a UI at all.

The Process: Layers, Not One Shot

The mistake most people make with AI prototyping is asking for the finished product in one message. I build in versions instead — logic first, UI last — so every version is something I can actually test before adding the next layer.

STAGE	FOCUS	WHAT I'M CHECKING
v1	Core logic	Does the calculation / flow / data parsing actually work, on real data?
v2	Controls & filters	Can a user actually steer the tool — inputs, toggles, filters?
v3	Core scenario	Does the main use case hold up end to end, not just the happy path?
v4–v5	Real feedback	What does a real user try to do that I didn't design for?
v6–v7	Polish & ship	Is it simple enough that someone opens it and just uses it?

WORKING PRINCIPLES

What Makes a Prototype Trustworthy, Not Just Fast

Real data from message one

I upload the actual file or describe the actual data shape immediately — not a sample. Fake data hides the messy edge cases (missing fields, odd formats, duplicates) that break a tool later. Better to hit them in v1 than v5.

Logic before styling, always

A prototype that calculates or flows correctly but looks plain is useful. A polished one with wrong numbers is dangerous — especially in front of investors or stakeholders. I don't ask for color, spacing, or animation until the underlying logic is verified.

Feedback by location, not vibe

"Make it better" wastes a round trip. I give AI the same kind of feedback I'd give a designer: what's wrong, exactly where it is, and what it should become instead — so the next version fixes the right thing on the first try.

Real users at v2 or v3, not v7

I show the prototype to a few real people well before it's finished. What they try to do that I didn't account for becomes the next version's priority — usually the most valuable feature in the whole build.

In Practice

v0 | Competitive teardown | Shipped as code

I ran a competitive UX/copy teardown of another PM's portfolio site, then fixed my own site's UX and copy directly through v0 — changes landed as working code, not a redline document someone else had to implement. The same logic-first, version-by-version approach carries into AGAR: interface concepts get prototyped as something a viewer can actually click through before a single design asset is finalized.

> *"A tool that's ugly but correct is useful. A tool that's beautiful but wrong is dangerous — so logic always ships before polish."*

REST OF THE TOOLKIT

Where AI Shows Up Elsewhere in My Process

Rapid prototyping is the core of how I work, but the same "define the structure, let AI fill volume, apply my own judgment" approach carries into the rest of the PM lifecycle.

PRDs, Briefly

IDEA → SPEC

I use AI to get from a rough idea to a structured PRD fast, so the first draft is a real starting point for review. Example: a full PRD and build prompt for SprintUp covering a complete microservices architecture.

User Story Simulation

SPEC → STRESS TEST

Before a feature reaches design or engineering, I have AI role-play the user against the user stories, surfacing friction or missing states early. Example: the MatchTrip PRD, simulated across four progressively tightening constraint stages.

Presentations & Dashboards

SPEC → STORY

For decks, I script content and layout as structured data so revising the narrative means editing data once. Example: AGAR's 13-slide investor deck, generated programmatically via Node.js/pptxgenjs.

The Common Thread

Every one of these — prototypes, PRDs, simulated user stories, decks — follows the same move: I define the structure, AI fills the volume within it, and my judgment decides what survives. The result isn't AI-generated work I ship as-is; it's raw material I filter, edit, and take full ownership of.

Yash Mahadik · Product Manager & Founder, AGAR · Mumbai, India · yashmahadik.vercel.app