A playbook by Saurav Kumar Nanda

The 0 → 1 Product Playbook

A field guide for founders and first-time product builders

Saurav Kumar Nanda

Product builder · Founder, Nandighosh Mobility · MS Tech Entrepreneurship, Illinois Tech

sauravnanda.com · hello@sauravnanda.com

Preface — Why this exists

I wrote this for the version of me who was staring at a blank Figma file in 2024, trying to turn an idea for an intercity bus company in Odisha into a real product. There are a thousand books on product management and almost none of them help when you are alone, with no team, no funding, and a customer waiting.

This playbook is the compressed version of everything I wish I had known when I started Nandighosh Mobility — and what I have since applied as Chief Sales Officer at Boomerangme, product lead with a Chicago stealth startup, and as a BI analyst at Flipkart before that. It is biased toward shipping. It will not teach you Scrum, OKRs, or any framework you have already heard of. It will teach you how to take an idea and turn it into something real people pay for, in weeks, not quarters.

Read it end-to-end once, then keep it open as a reference. Every chapter has a checklist at the end — use them literally.

> This is a living document. You can reach me at hello@sauravnanda.com with corrections, additions, or war stories of your own.

CHAPTER 01

Customer Discovery

You cannot build a product for a customer you have not yet met.

The biggest mistake first-time founders make is not building the wrong thing. It is building before they have earned the right to know what the right thing is. Customer discovery is that earning process.

The target is simple: 30 interviews with real people in your candidate ICP, inside 30 days. Not surveys. Not social posts. Not focus groups. Thirty voice-to-voice conversations, 25 minutes each, recorded with permission. This is the single highest-leverage thing you can do in month one.

Who to talk to

Cast wider than you think you should. If you believe your buyer is a VP of Engineering at a mid-market SaaS company, talk to five VPs — and also five Directors of Engineering, five Heads of Product, and five founders of similar-sized companies. The pattern that matters is not the title; it is the pain.

A short ranking of recruitment channels by cost and signal quality:

Channel	Cost	Signal quality	Notes
Warm intros	Free	High	Best for first 10 interviews
LinkedIn cold outreach	Free	Medium-High	30-40% reply if the message is specific
Founder communities (Slack, Reddit)	Free	Medium	Selection bias toward early adopters
Intro.co / Pickle	$50-150/call	High	Worth it for hard-to-reach enterprise buyers
Event / conference halls	Travel cost	Medium	Ten brief chats can match one deep call
Cold email to named list	Free	Medium	Works if the list is tight and personalized

Booking the call

The request is the hardest part. Here is the message that has worked for me across three very different products. It is short, specific, honest about what you are doing, and asks for a clear amount of time.

> *“Hi [name] — I'm Saurav, building a [one-line product description] for [specific segment]. I'm not selling anything. I'm doing 30 discovery calls to understand how people actually solve [specific problem today], and I'd love to learn from you. 25 minutes on Zoom next week? I promise no pitch, and I'll share what I find back.”*

Notice what this does: it names a specific segment, it anchors to the problem (not the solution), it promises a clear time cap, and it offers reciprocity in the form of the research findings. Reply rates on this template have been 35% to 55% across my last three products.

The interview itself

The mistake everyone makes is talking too much. You are there to listen. I aim for 80% customer talk time. The script I use has four moves and I keep it under a single page:

1. Open with gratitude and a reminder: "I'm not selling anything today. I'm trying to learn how you actually do X."
2. Ask them to walk you through the last time they faced the problem, step by step. Concrete, recent, specific.
3. Find the workaround. How are they solving this today? What tools, what people, what manual steps?
4. Ask about the emotional cost. What does it feel like when it goes wrong? When is the pain worst?

The questions that unlock the most signal, in my experience:

- Walk me through the last time you [did X]. What happened?
- What was the hardest part? Why?
- What have you tried already? Why did you stop?
- If you had a magic wand, what would the ideal version look like?
- How much time or money does this cost you per month?
- If I built [wedge], what would stop you from using it?

Never ask "would you use this?" or "would you pay for this?" — you will get lies, and kind ones. Ask about behavior, not intent.

Objection handles

A few objections come up on nearly every call. Memorize responses:

Objection	Response
"I don't have time right now."	"I understand — 10 minutes to share the one thing that frustrates you most would still help me. Is tomorrow 8:30 okay?"
"We already use [competitor]."	"That's exactly why I want to talk. What's working? What's breaking? I'll share back what others in your shoes are doing."
"I'm not the right person."	"Who on your team lives this problem day to day? A warm intro would help enormously."
"Can you send a survey instead?"	"I've tried that — surveys miss the why. 25 minutes on Zoom saves me 10 surveys."

Synthesis — from raw notes to themes

After every call, take five minutes to write a short after-action: the quote that stung, the workaround they admitted to, their rating of urgency (1-5), and whether they asked when you would launch. These five minutes compound. After 30 calls you will have 30 tight summaries, not 30 one-hour transcripts to wade through.

At call 15 and call 30, do a synthesis pass. Open a sheet or whiteboard, write every pain point on a sticky, cluster them, and count. I use a simple rule: if a pain shows up in 40% or more of interviews with consistent workaround behavior, it is a real pattern. Below 20% and it is noise. Between is ambiguous — investigate further.

Chapter 1 checklist

- 30 interviews booked and completed within 30 days
- Each call 25 minutes, recorded with permission
- Five-minute after-action written within 30 minutes of the call
- Synthesis done at interview 15 and interview 30
- At least one pain point with 40%+ frequency identified
- Candidate wedge defined (one customer, one problem)

CHAPTER 02

Writing the PRD

A PRD is a contract with your team. Not a pitch deck.

A product requirements document — PRD — is the most misunderstood artifact in a startup. People confuse it with a pitch deck (persuading), a press release (marketing), or a spec (listing everything). It is none of those. It is a contract between you and the people building with you, about what you are going to ship, why, and what "done" looks like.

If your PRD is longer than two pages, you have either not thought hard enough or you are hiding behind words. The best PRDs I have written are one page. The four-part template below has held up across Nandighosh (offline product), a Chicago SaaS, and Boomerangme's loyalty platform.

The four-part template

1. Why now. Two sentences. What changed in the world, the customer, or the data that makes this the right thing to build this quarter?
2. Who for. One sentence. The specific user and moment — not the segment. "A first-time SaaS founder in month 3, stuck on pricing" beats "early-stage founders."
3. What it does. Three to five bullets. User-visible behavior, not implementation. "User uploads a CSV and sees top themes in under 30 seconds" — not "we integrate LangChain."
4. How we'll know it worked. One metric, one threshold, one deadline. "60% of new users reach first value in under 5 minutes, measured by activation funnel, by end of month 2."

> Every PRD I have shipped in the last two years fits on a single page. If you cannot do that, the decision is not ready.

What does NOT belong in a PRD

- Market size / TAM (that's for the deck)

- Technical architecture (that's for a design doc)
- Sprint breakdowns (that's for the eng lead)
- Competitor analysis (that's background, not requirement)
- Edge case enumeration longer than the happy path

Annotated example — the PRD for the Nandighosh booking flow

> Real PRD I wrote in January 2024 before a single line of code was written for our bus booking stack.

Why now: Odisha bus travel is still dominated by walk-in ticketing at the station. 70% of our pilot passengers booked by calling the office. We can 10x reach by letting people book from their phones before arrival.

Who for: A working-class passenger in Odisha, aged 22-45, with a feature phone or basic Android smartphone, who has never booked bus travel online before.

What it does: (a) User opens a WhatsApp link, sees upcoming routes; (b) picks a seat visually; (c) pays via UPI; (d) receives a ticket SMS with QR; (e) shows QR at boarding.

How we'll know it worked: 40% of bookings on route R1 come via WhatsApp flow, measured at 60 days post-launch. At 30% we iterate the UX. Below 20% we kill and revert to call-in.

Total length: 127 words. The build took three weeks. Launch was on time. Actual adoption at day 60 was 47%. We kept it and scaled.

Chapter 2 checklist

- PRD fits on one page
- Why now answers a change in world, customer, or data
- Who for is a specific user in a specific moment
- What it does is 3-5 user-visible bullets
- Exactly one success metric, one threshold, one deadline
- Every bullet survives the "so what?" test

CHAPTER 03

MVP Scoping

The question is not what to build. It is what to leave out.

Most MVP failures I have seen are scoping failures. The founder tries to ship the full vision on day one, the team grinds for three months, nothing gets in front of customers, and the window closes. MVP scoping is a discipline of subtraction.

MoSCoW, honestly applied

MoSCoW — Must, Should, Could, Won't — is a cliché, but the honesty it forces is not. The rule I apply: if you put more than 8 items in "Must," you are not doing MoSCoW, you are just making a backlog. Here is how I use it:

Tier	Definition	How many items
Must	User cannot reach first value without it	5-8
Should	Ships in v1.1, within 30 days of v1	6-10
Could	Parking lot. Revisit at quarterly planning	Unbounded
Won't	Explicitly killed, with reason documented	Write 3-5

The cut-first column

For every feature in Must and Should, add a column: "If the timeline slips by a week, do we cut this?" Force a Yes / No. If you cannot answer, the feature is not scoped. A team that pre-commits what it will cut ships twice as fast as a team that negotiates scope under pressure.

Design partners — why you want 3, not 1

A design partner is a customer who has agreed to use your MVP pre-launch, give detailed feedback, and accept that things will be broken. At Nandighosh our design partner was a local textile factory that needed to move 60 workers between two cities on Mondays. At the Chicago startup we had three design partners across three ICPs.

Three is the right number. One gives you a product built for one customer — dangerous. Five means you cannot incorporate all the feedback and someone will be angry. Three forces you to distinguish signal (everyone asks for it) from noise (one asks for it).

> Design partners are not free beta users. They are contracted relationships: they commit to weekly feedback, you commit to weekly shipping. Put it in writing.

Chapter 3 checklist

- Must list is 5-8 items
- Every Should item has a cut-first answer
- At least 3 design partners recruited before build starts
- "Won't" list written with reason
- Effort-to-Must ratio below 70% of available runway
- Weekly demo cadence with design partners locked in

CHAPTER 04

Launch & Loops

A launch is not an event. It is the first data point of a loop you will run forever.

I have watched a lot of founders treat launch day like a finish line. They shine the press release, tweet the Product Hunt link, and crash. Launch is a starting line. The interesting work is the loop you run in weeks one through four — because that is where you learn whether you actually have something.

Pre-launch — the 2-week runway

Two weeks before launch, do five things:

- Freeze the feature set. No new "just one more thing."
- Write the launch email (to waitlist, design partners, and network).
- Set up analytics — the 3-5 metrics you will stare at daily. Not 30.
- Brief your design partners. They are your day-0 testers.
- Draft the launch week schedule: when to post, who to email, what to follow up on.

Day 0 — the boring version

The launches that matter are boring from the outside. The team sees the number-one metric go up (or not) and knows what to do next. That is the target. Skip the livestream. Skip the hype. What matters on day 0:

- First 10 real signups (not from friends)
- First user to reach activation (your defined "first value")
- First friction log — every place a user got stuck
- First revenue, if applicable

Week 1 — the raw numbers

By end of day 7 you should know, to the user, each of these:

Metric	What it tells you
Signups (count)	Is the message landing?
Activation rate (%)	Is onboarding working?
Time to first value (median)	Is the product clear?
Day-1 retention (%)	Did anyone care enough to come back?
Day-7 retention (%)	Is there a real need underneath?
Top friction (qualitative)	What to fix this week?

Week 4 — the first honest review

After 28 days, do a cold review. Pull the metrics, look at the activation curve, read every support ticket, and write a one-page answer to three questions:

1. Is anyone coming back? (retention curve)
2. When they come back, why? (feature usage)
3. When they do not come back, why not? (friction log + exit interviews)

> Week-4 is the most honest you will ever be about your product. Do not skip this pass. Write it down, share it, keep the file.

Chapter 4 checklist

- Feature freeze 14 days pre-launch
- Launch email drafted and scheduled
- 3-5 metrics instrumented and visible before launch
- Week-1 numbers reviewed live on day 7
- Week-4 one-page review written and shared
- Five user interviews with active users before day 30

CHAPTER 05

The First 90 Days of Metrics

What to track. What to ignore. How to stay sane.

Product metrics can either run your team or ruin it. In the first 90 days, less is more. I have seen founders build 40-chart dashboards and make zero decisions from them. The goal is the opposite: five numbers you can recite from memory and walk into any meeting with.

The five numbers

1. Activation rate. % of new signups who reach first value within the defined window.
2. Week-1 retention. % of signups who return within 7 days.
3. Week-4 retention. % of signups who return within 28 days. This is your north-star proxy.
4. Revenue per active customer. Even if free, track intent-to-pay signals.
5. Top friction. A qualitative, weekly-updated ranked list of three user blockers.

> If you cannot recite these five numbers right now, without opening a dashboard, you are not running the product — the product is running you.

What to ignore in days 1-90

- Vanity traffic metrics (pageviews, impressions)
- NPS — too early to be meaningful with small n
- Cohort LTV — cohorts are not mature enough
- Conversion funnel micro-optimization — fix big leaks first
- Competitor dashboards — a distraction, not a driver

The weekly operating cadence

Every Monday, 30 minutes, five beats:

1. 0-5 min: read the five numbers aloud. No commentary.
2. 5-12 min: what shipped last week? (facts, not spin)
3. 12-20 min: what ships this week? (one-sentence per person)
4. 20-25 min: top risk — the one thing that could blow up
5. 25-30 min: write notes. Close laptop. Ship.

Chapter 5 checklist

- Five-number scorecard defined and visible daily
- Weekly operating review established on a fixed day/time
- Top-friction list updated every Monday
- 30-60-90 review scheduled before launch
- No more than 8 metrics on any dashboard
- Every metric has a named owner

CHAPTER 06

Team of Two or Three

The first hires decide the culture. The culture decides the company.

Most early-stage products are built by one or two people for the first 6-9 months. By the time you reach product-market fit, the team is usually three to five. Those first two hires — after the founding team — are the most important product decisions you will make. I have watched great products die because the founder hired the wrong second engineer.

The first hire — a shipper, not a visionary

Before the money is in, and sometimes after, you want someone who will ship at 90% quality three times faster than you. Not someone with a vision. You already have one. The person you need is the one who closes tabs, merges PRs, and ends days with deployable work. At Nandighosh my first operations hire had never worked at a startup — he had run a tire shop for 12 years. He shipped more in a week than three MBAs would in a quarter.

The second hire — the balance

The second hire should be the axis orthogonal to the first. If your first is a back-end engineer, the second is design or customer-facing. If your first is a salesperson, the second is an engineer. The rule: at two hires you should be able to do the full customer loop — talk to them, build for them, ship to them — without waiting on an external person.

A practical interview loop

1. 30-min screen — check basic fit, attitude, compensation expectations
2. 90-min craft interview — a paid take-home or paired build of something representative
3. 60-min founder interview — walk them through the roadmap, see what they push back on
4. Reverse-interview — they interview you. Do they ask about product, or only compensation?

5 Reference-check two people who reported to them, not just their managers.

Compensation — clean, cheap, and honest

Early-stage salary: 60-80% of market, with meaningful equity vested over 4 years with a 1-year cliff. Do not do tricky vesting. Do not do non-competes. Do not promise future raises verbally. Write it, sign it, and move on. Bad compensation practices are the single fastest way to poison an early team.

> I have regretted every hire I made in a hurry. I have never regretted a hire I was patient with.

Chapter 6 checklist

- First hire ships more than they talk
- Second hire covers the axis you do not
- Standard interview loop, no exceptions
- Offers written, equity cleanly documented
- No surprise compensation variance across the first 5 people
- Two reference calls per hire, minimum

CHAPTER 07

The Failure Modes

Most 0→1 efforts fail in one of five predictable ways. Know them cold.

After watching 30+ 0→1 efforts up close — my own, portfolio companies, friends' — I have come to believe that failure is not idiosyncratic. It looks personal when it happens to you, but patterns repeat. If you internalize these five, you will catch yourself.

Failure mode 1 — feature creep before first value

The founder keeps adding features because no feature seems "complete enough" to launch. Three months in, no user has touched the product. Fix: set a hard date 8 weeks from today. Cut 40% of the scope, ship to 5 design partners. Feature creep is usually fear of judgment in disguise.

Failure mode 2 — building for oneself

The founder is their own most available customer, so they build for their own workflow. The problem: they are not representative. I made this mistake with Nandighosh's first app UI — built it for a smartphone-native user, but 70% of our target was feature-phone-literate. Fix: sit next to three real target users as they try to complete a real task. Watch them fail. Rebuild.

Failure mode 3 — selling too early

The founder raises money on a vision, feels pressure to show revenue, sells to anyone who will take a call, and discovers six months later that those customers are the wrong ICP. Churn kills them. Fix: pick the ICP before the pitch deck. Say no to out-of-ICP deals even if it hurts.

Failure mode 4 — hiring before shipping

The founder raises a seed round, hires four people in month one, and now runs a company of five people who collectively have not shipped anything. Fix: hire reactively. Hire number N+1 only when person N has shipped their second thing. Keep the team small until velocity forces expansion.

Failure mode 5 — metrics theater

The founder builds a 30-chart dashboard, presents it weekly, but makes no decisions from it. Fix: burn the dashboard, pick 5 numbers, recite them daily. A product is run on decisions, not charts.

> Every one of these five has happened to me personally at least once. The point is not to avoid them — it is to recognize them within 30 days, not 6 months.

Chapter 7 checklist

- Weekly self-audit: am I adding features or shipping to users?
- At least two users not like me inside the first 10 design partners
- ICP is one sentence and defensible
- Hiring plan tied to shipped milestones, not calendar dates
- Five-number scorecard visible, no 30-chart dashboard

CHAPTER 08

Pricing and Packaging for v1

The price you pick on launch day is a diagnostic, not a prediction.

Pricing is the scariest decision in the first 90 days because it is visible, reversible only with pain, and it silently selects your customer base. Founders typically pick a price by vibe — then spend months wondering why their customers are wrong for the product. The price made the customer.

Three honest approaches for v1

Approach	Best for	Risk
Price high, discount later	High-touch enterprise, design partner stage	Scares off self-serve users
Price low, raise on evidence	Self-serve SaaS, high volume	Hard to raise on existing customers later
Skip pricing, partner first	Deeply unproven wedge, no comparables	Misses the willingness-to-pay data

The Boomerangme pricing story

At Boomerangme we launched the SMB plan at $49/month. No one blinked. Within 60 days we raised to $99. Still no one blinked. We tested $199 and close rates finally dropped 20%. That told us where the SMB price ceiling was. We settled at $149 — the right price was discovered, not predicted. The lesson: if nobody blinks, you are leaving money and signal on the table.

Packaging moves to try in v1

1. Flat monthly with a 30-day trial — easiest to pitch, easiest to cancel
2. Annual with a 15-20% discount — improves cash and churn, adds friction at first close

3. Usage-tiered (per seat, per call, per volume) — better fit for usage-based pain, harder to predict
4. Free tier + paid upgrade — only if you can generate a strong free-to-paid path (most teams cannot, early)
5. Founding customer pricing — "first 10 customers, locked rate for 24 months" — creates urgency and converts discovery partners

> Do not design pricing as if it were permanent. Every v1 price is a hypothesis. The job is to ship a hypothesis you can falsify, then iterate.

Annual vs. monthly — a small-team perspective

At a pre-seed or seed company, cash matters more than elegance. Annual contracts pull a year of revenue forward at a 15-20% discount. If you close ten customers at $10k ARR, annual vs. monthly is the difference between $100k in the bank today vs. $100k arriving over 12 months. That bought me three extra engineering months once.

Raising price on existing customers

You will want to raise prices. Rules I follow: never raise on a customer inside their current term; give 60 days notice; grandfather any customer who has paid on time for 12+ months at the old rate for another 12; introduce new pricing on new customers first. I have never lost a grandfathered customer over a price raise.

Chapter 8 checklist

- v1 price picked with an explicit hypothesis
- Pricing review every 90 days for the first year
- Annual plan available by day 30 post-launch
- Grandfathering rule written before first price raise
- No more than 3 pricing tiers in v1

CHAPTER 09

Support as a Product Channel

In the first 12 months, support is where the product is actually being built.

Founders typically think of support as a cost center to be automated away. In the first year, it is the opposite. Every support ticket is a signal about the product. Treat it like product feedback with a receipt. The teams that win early are the ones that read every ticket personally for the first year.

The founder-does-support phase

For the first 200 customers, one of the founders should see every support ticket. This is non-negotiable. At Nandighosh, I personally handled booking escalations for six months — it taught me more about the real product than any dashboard. At a Chicago startup I advised, the founder refused this and shipped the wrong fix three times in a row.

The ticket-to-product pipeline

1. Tag every ticket with a product area and a severity
2. Weekly: review the top 3 areas by ticket volume
3. Monthly: turn the top 5 ticket types into a one-page fix spec
4. Quarterly: review which fixes moved the ticket volume, which did not

What support signals predict

- Spikes in onboarding tickets = broken first-time experience
- Spikes in billing tickets = unclear pricing or integration issue
- Spikes in "how do I..." tickets = missing discoverability in the UI
- Spikes in "this does not work" = actual regression
- Silence in a feature area = either perfect or unused; check usage data

> The best product telemetry at seed stage is not analytics. It is a well-read inbox. Founders who ignore this run blind.

Chapter 9 checklist

- Founder reads every ticket for first 200 customers
- Tickets tagged by area + severity
- Weekly review of top-ticket areas
- One ticket-driven fix shipped every sprint
- First support hire only after tag system is trusted

CHAPTER 10

Writing Is the Second Product

What the team writes shapes what the team ships.

Every early-stage product team has two products: the thing they ship to customers, and the thing they write to each other. Memos, PRDs, docs, Slack threads, decision logs, readme files. Most teams invest heavily in the first and ignore the second. The teams that take writing seriously move faster, onboard faster, and change course more cleanly than teams that don't.

Why writing beats talking

A meeting of five people costs five hours of total attention. A one-page memo takes 30 minutes to write and 5 minutes to read. On attention alone, writing is a 30x efficiency gain. On quality, it is higher still — writing forces the author to think, and the reader to absorb at their own pace. The most expensive meetings I sit in are ones where a memo would have sufficed.

The four documents that run my teams

1. The PRD — one page, one decision, one metric (see Chapter 2)
2. The weekly memo — 200 words every Monday, sent to all-hands, covering what shipped, what ships, what is stuck
3. The decision log — a running list of non-obvious calls, dated, with the rationale and the person responsible
4. The postmortem — whenever something breaks, one page within 48 hours: what happened, what we learned, what we changed

Writing quality — rules I hold myself to

- Every sentence earns the next
- No adjectives without numbers

- Use I, not we, when attributing decisions
- Use we, not I, when attributing outcomes
- Kill every word that does not change meaning if removed
- Write the one sentence you would highlight if you were the reader

> A team that writes well is a team that thinks well. A team that refuses to write is a team that will re-learn the same lessons forever.

The weekly memo template

Monday morning, 200 words, sent before 10am:

1. One sentence: the headline metric, this week vs last
2. Three bullets: what shipped last week
3. Three bullets: what ships this week
4. One bullet: the one thing I need help with

Chapter 10 checklist

- Weekly memo shipped by Monday 10am, no exceptions
- PRD template enforced; no multi-page PRDs
- Decision log active and scanned weekly
- Postmortem written within 48 hours of every meaningful incident
- Writing standards read and understood by every hire in first week

CHAPTER 11

A Letter to the Version of You Reading This

If you only read one chapter, read this.

If you have read this far, you are probably sitting with one of three thoughts: a) I am about to start, how do I begin? b) I have started and I am stuck. c) I am building and I want to get better. Here is a short, direct note for each.

If you are about to start

Pick the smallest possible version of the problem. Talk to ten people who have it — this week. Write a one-page PRD. Ship something ugly in 30 days. The rest is commentary. The only mistake you can make at the start is planning instead of moving. Plans will be wrong. Movement will be right, eventually.

If you are stuck

Stuckness is almost always a signal that you have too many open questions at once. Write them all down. Force-rank them. Pick the top one. Kill or defer the bottom three. I have run this ritual twice at Nandighosh — both times the answer was to cut scope ruthlessly and restart from the smallest customer commitment we still believed in.

If you are already building

The book's last 10 pages are the most applicable to you. Chapter 8 on pricing, Chapter 9 on support, Chapter 10 on writing. Pick one. Install one new habit. Leave the others until the next quarterly review. Speed kills habit change; patience compounds it.

Every playbook in this collection is useful. None of them matter if you do not ship. Close the PDF. Open your inbox. Send the message. Start.

— Saurav Kumar Nanda · Chicago & Balasore · 2026

CHAPTER 12

Closing

Ship. Then ship again.

The honest secret of 0→1 is that nothing in this book is complicated. Most founders know most of it. The difference between the ones who ship and the ones who do not is the willingness to act imperfectly on incomplete information, every day, for 18 months.

If you take one thing from this playbook, take this: the best product strategy is a shipped one. A half-built thing in front of a real customer teaches you more than six months of research. The frameworks in here are useful — but only as scaffolding around the act of shipping.

If you use anything in this playbook and it moves the needle, please tell me at hello@sauravnanda.com. I read every email.

— Saurav Kumar Nanda · Chicago & Balasore · 2026