— AI-FIRST PRODUCT DEVELOPMENT

Planning, Prompting & Shipping With AI

One real product, from idea to shipped — the AI-first way.

Presented by **Derek Peters**

50 minutes · self-contained · One product, idea → ship · Every prompt & output shown

Gauntlet

— TONIGHT'S RUN OF SHOW

One product, idea to ship

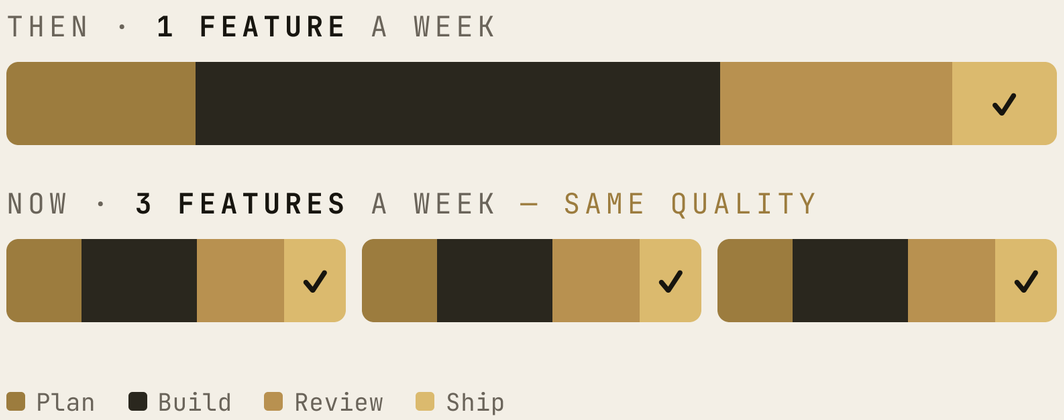
01	**Planning** A vision doc, a PRD (Product Requirements Document), and a roadmap the AI can build from	~16 min
02	**Prompting** Drive the build straight out of the documents	~15 min
03	**Shipping** Release, measure, and close the loop	~12 min
—	**Recap & Q&A**	~7 min

Gauntlet

— THE PREMISE

Your job just changed jobs

- **The bottleneck moved.** Typing code is cheap now — deciding what to build and proving it works is the job.
- **AI multiplies judgment** — including bad judgment. It's never been easier to vibe-code the wrong thing, fast.
- **The leverage isn't the prompt.** It's the planning before it and the shipping after it.

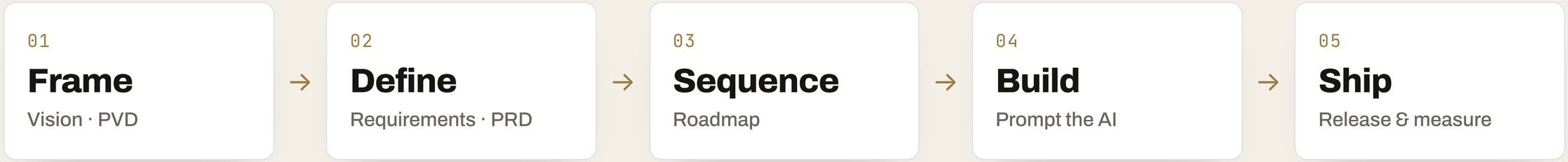


Same clock, same quality — AI just fits more cycles in it.

Gauntlet

— THE METHODOLOGY

The AI-First Product Loop

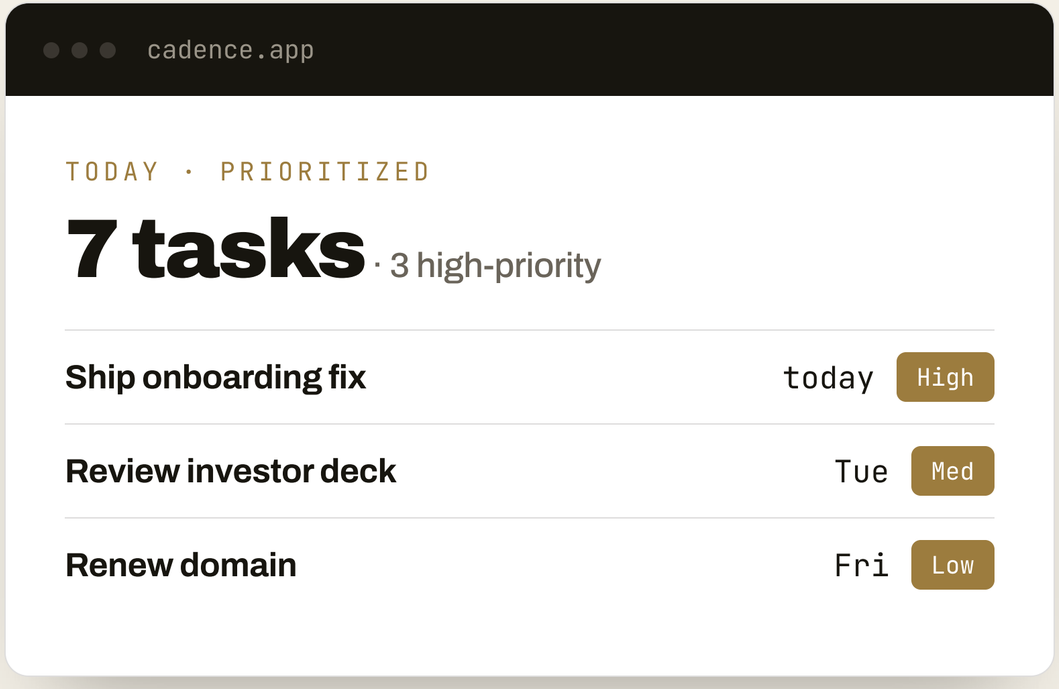


PVD = Product Vision Document · **PRD** = Product Requirements Document

Gauntlet

Our example: Cadence

- **Cadence is a task tracker with AI built in** — paste a brain-dump, get a prioritized list.
- **We'll trace its first feature** idea-to-shipped — you'll see every prompt and its output.
- **Real teeth:** messy human input, and an AI ranking people have to trust.



— WHY THE WORKFLOW

Two ways to build with AI

VIBE CODING

Riff till it runs

START	“Build me a task app” — and see what comes back
PATH	Chase whatever the model returns, prompt by prompt
RESULT	**Fast demo, fragile product** — you can't say why it works, or fix it when it breaks

VS

PLANNING-FIRST

Plan, then prompt

START	Vision → PRD → roadmap — decide the bet before you build
PATH	AI builds from documents you own and can steer
RESULT	**Same speed, but it compounds** — you know what's done and what's next

Same models, opposite outcomes — the gap is the **workflow**. So before the first prompt, here's the kit we run it on.

— THE TOOLKIT

What we actually use — per stage

01 Planning

MODEL

- **Frontier reasoning model** — Claude Opus, GPT-5.5, Gemini

WHERE

- **Claude Desktop / ChatGPT** — Projects
- **Docs in Markdown / Notion**

OUTPUT

- **PVD · PRD · roadmap**

02 Prompting

AGENT

- **Claude Code · OpenAI Codex**
- **Cursor** — Copilot / Windsurf

MODEL

- **Claude Sonnet / Opus**
- **GPT-5.5-Codex** — fast model for loops

CONTEXT

- **CLAUDE.md / AGENTS.md**

03 Shipping

PIPELINE

- **GitHub Actions · flags**

AI REVIEW

- **Codex review · CodeRabbit**
- **Greptile**

TEST & MEASURE

- **Vitest · Playwright**
- **PostHog / Amplitude**

Pick one per row — the workflow matters more than the brand. Reasoning model to plan, fast model to build.

— SECTION 01

Planning

01

Frame, define, sequence. Three documents before a single prompt — the plan is the product.

— PLANNING · FRAME

Start with a vision document

- **One page:** why this exists, for whom, and the one metric that matters.
- **Don't ask AI for a vision** — make it interview you until yours is sharp.
- **Principles are guardrails.** For a daily tool: calm, trustworthy, you stay in control.

```
prompt ▸ "Act as a product coach. Interview me one question
         at a time about Cadence: the user, the core problem,
         the one metric that matters, and three principles.
         Then draft a one-page PVD. Don't invent answers — if
         I'm vague, push back until it's sharp."
```

PVD · CADENCE

vision	**Turn the chaos in your head** — into a clear, prioritized plan
for	**Makers drowning in scattered to-dos**
problem	**Tasks live everywhere; prioritizing is manual**
north star	**Planned tasks completed / week**
principles	**Calm · Trustworthy priority · In control**

— PLANNING · DEFINE

Turn vision into a PRD

- **The "what," for v1 only** — problem, user stories, and hard scope lines.
- **Out-of-scope is the gold.** It's how you stop the AI from gold-plating.
- **Make "done" measurable** so a human and the AI can both check it.

```
prompt ▸ "From PVD.md, draft a v1-only PRD. Include: problem,
         3–5 user stories (as a… / I want… / so that…), in-
         scope vs out-of-scope, and measurable success. Cut
         what isn't needed to prove the one bet that matters
         — that people trust the AI's priorities; park the
         rest as out-of-scope. Ask before assuming."
```

PRD · EXTRACT & PRIORITIZE · V1

PROBLEM

Tasks are scattered; ranking them is constant manual work

USER STORIES

- **Paste a brain-dump of everything**
- **Get back clean, prioritized tasks**
- **Edit any priority or due date**

IN SCOPE	OUT OF SCOPE
Capture · extract · prioritize · edit	**Teams · integrations · auto-complete**

SUCCESS

Prioritized list in < 5s · tasks completed

— PLANNING · SEQUENCE

Sequence with a roadmap

Not a wish-list — sequence by **risk and value**. Everything in **Now** exists to kill the riskiest assumption first.

```
prompt ▸ "From PRD-v1, build a Now / Next / Later roadmap. Sequence by
         risk × value, not by excitement. 'Now' = only what tests the
         riskiest assumption — do users trust the AI's priorities?
         One line of rationale per item; keep Next and Later
         deliberately loose."
```

NOW	NEXT	LATER
Brain-dump capture	Due-date nudges	Team boards
AI extract & prioritize	Quick-add & edit	Slack & email capture
Editable task list	Calendar sync	AI daily digest

Biggest unknown to de-risk: will the AI's priorities feel right enough to trust?

Make the docs the AI's context

- **Fold PVD, PRD, and roadmap** into one file the AI reads every single time.
- **Capture stack, conventions,** and a one-sentence definition of "done."
- **Treat context as code** — version it, review it, keep it current.

CONTEXT.md

```
# the AI reads this every time
product:  Cadence — see PVD.md
build:    PRD-extract-v1.md  ← this slice
roadmap:  Now → capture & organize

## Stack & conventions
- Next.js · TypeScript · Tailwind
- typed AI output (zod) · feature-flagged
- analytics on every action

## Definition of done
Paste a brain-dump → prioritized,
editable task list in < 5s.
```

— SECTION 02

Prompting

02

Drive the build like a tech lead. The documents you just wrote are the prompt.

Prompt like a tech lead

- **Intent, constraints, acceptance** — all three in one breath.
- **Pull rules from the PRD;** don't make the model invent them.
- **Ask for a plan first;** approve it before a line of code is written.

ANATOMY OF A GOOD ASK

intent	What & why, in one sentence
context	Files, stack, patterns to follow
rules	Straight from the PRD & CONTEXT.md
accept	The PRD's own success metric

Give the model the map

- **Point at exact files** and an existing pattern to follow.
- **The PRD is the prompt** — you're just feeding it back the docs.
- **End every ask** with "plan first, no code until I approve."

prompt.txt

```
Implement "Extract & Prioritize" (PRD-v1).

# Context — the docs are the spec
read:   PRD-extract-v1.md · CONTEXT.md
edit:   src/extract.ts · ui/TaskList.tsx
follow: src/ai/client.ts

# Rules (from the PRD)
- return typed JSON {title, priority, due?}
- priority from urgency; never invent a date
- validate with zod; fail safe on bad output

# Done when
- brain-dump → editable list in < 5s
- tests green: happy, empty, off-schema

Plan first. No code until I approve.
```

A bad prompt vs a good one

● Bad — vague, no rules

prompt

```
add an AI feature that turns
notes into tasks. make it smart.
```

× WHAT YOU GET

Splits text on line breaks. No priorities, no due dates, a schema that changes every run — and no way to edit it.

● Good — the PRD, handed back

prompt

```
Implement "Extract & Prioritize" (PRD-v1).
read:   PRD-extract-v1.md · CONTEXT.md
edit:   src/extract.ts · ui/TaskList.tsx
return typed JSON {title, priority, due?}
accept: brain-dump → editable list in < 5s
Plan first. No code until I approve.
```

✓ WHAT YOU GET

HIGH	Ship onboarding fix	today
MED	Review investor deck	Tue
LOW	Renew domain	Fri

+ a short plan you approve first

Iterate in tight loops

- **Small diffs over big rewrites** — one change you can verify.
- **Read every line.** You own the code, not the model.
- **When it's stuck,** reset the context instead of arguing with it.

Four ways it falls apart

- **Every one is a process failure** — not a model failure.
- **The cure is always the same:** smaller asks, read the diff, real acceptance tests.

SMELL	FIX
600-line diff you can't review	One slice per prompt
Confident code, quietly wrong	Read every line; run the test
Arguing with a stuck model	Reset the context, start clean
Shipping on "looks done"	Define done as a test

— SECTION 03

Shipping

03

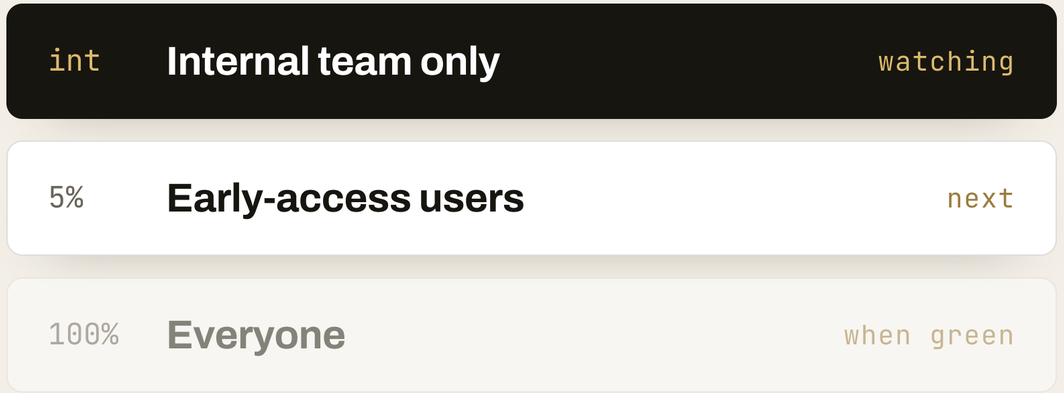
Get it to real users without holding your breath. Small, flagged, measured — then learn.

— SHIPPING

Ship behind a flag

- **Merge small, merge often** — flags let unfinished work land safely.
- **Keep main releasable** at all times.
- **Roll out gradually;** a switch beats an emergency deploy.

```
prompt ▸ "Wrap the extractor in flag cadence_extract_v1,
          default off, with a fallback to manual entry. Roll
          out internal → 5% → 100%, gated on error rate < 1%."
```



Let AI watch the build

- **Generate tests alongside** the feature — not after.
- **Wire AI into CI** (continuous integration) for review, not just authoring.
- **The pipeline is the gate:** green, or it doesn't ship.

```
prompt ▸ "Generate tests with the feature — happy path, empty
         dump, and an off-schema response that must fail
         safe, not crash. Make them a required CI gate on
         every PR."
```

.github/workflows/ship.yml

```yaml
on: [pull_request]
jobs:
  guard:
    steps:
      - run: pnpm test --coverage
      - run: pnpm typecheck
      - uses: ai/review@v2  # reads the diff
        with:
          fail_on: [secret, n+1, missing_test]

# green, or it doesn't ship
```

— SHIPPING

Measure, then close the loop

- **Instrument before you announce.** Dark launches teach you nothing.
- **Let the data pick** the next slice — not your gut.
- **Feed it home** — production learnings update the PVD and PRD.

```
prompt ▸ "Instrument before launch: track dump_pasted,
         tasks_extracted, priority_edited. Set a target
         (activation > 50%), pipe to PostHog, review weekly."
```

PRODUCTION SIGNAL · WEEK 1

activation	**61% paste a brain-dump** target was 50%
organized	**6.4 tasks extracted / session**
edits	**34% tweak a priority**
next	**→ "why this priority" explainer**

— THE LOOP, CLOSED

Plan → Prompt → Ship → Learn

01

Plan

PVD, PRD, and roadmap — the documents the AI builds from.

The docs are the spec.

02

Prompt

Build straight from the PRD. Plan first, tight loops, read every line.

You're the tech lead.

03

Ship

Flagged, gated by CI, measured — then fed back into the vision.

Green, then learn.

The throughline: **your PVD, PRD, and roadmap are the prompt — the product is the proof.**

— START MONDAY

Three moves you can make tomorrow

01

Write a one-page PVD

For whatever you're building right now — who it's for, the one job, the one metric.

02

Turn it into a v1 PRD

Scope it down hard, and write the "out of scope" list — that's what keeps you shippable.

03

Make the AI plan first

Before it writes a single line, have it propose a plan you approve.

No new tools required — **just the discipline to do the bookends.**

— GO FURTHER

Want to build like this every week?

This is the line between **vibe-coding a demo** and shipping a product that matters — and Gauntlet is where you cross it, every week, under real production pressure.

PRODUCTION ONLY

Every week requires shipped, working systems

WEEKLY LIVE REVIEWS

You demo deployed systems under scrutiny

ESCALATING COMPLEXITY

The bar rises every single week

Become a Challenger →

gauntletai.com/apply · 10 weeks, full-time, covered

Gauntlet

— OPEN FLOOR

Questions?

One product, idea to shipped — same loop, real documents, real prompts.

Gauntlet