AI Makes Products Easy to Build; PMs Now Own Selection, Distribution, and Proof

PM Daily Digest

2026-09-20

AI Makes Products Easy to Build; PMs Now Own Selection, Distribution, and Proof

By PM Daily Digest • September 20, 2026

Current signals show a new PM operating problem: choose what deserves to exist, make it discoverable through agents, and prove it works in real user workflows. The brief translates that shift into validation, execution, career, and tooling practices.

Big Ideas

The bottleneck is moving from building to selection and distribution. AI has expanded the supply of working software while the number of buyers and the time they have to evaluate products have not kept pace. When many products chase one buyer, being found— increasingly by agents as well as people—matters before technical superiority; with prototypes falling from weeks to minutes, judgment is the ability to discard weak versions quickly and concentrate on the one buyers value. [1] A parallel product-leadership argument is that AI makes it as easy to build the wrong product as the right one, so strategy and discovery decide what earns a place in the portfolio. [2] The operating change for PMs: make distribution evidence and ruthless selection gates part of product development, not post-launch concerns.

Consumer AI may split into cheap, narrow jobs and premium general capability. Andrew Chen argues that a model initially more than 400× cheaper than a general LLM could make ad-supported, free AI-native apps viable, but through differentiated point solutions rather than a generic assistant. His examples are 80/20 jobs such as filtering important email, finding dates, and prioritizing contacts, with general models added for higher-value "wow" moments. [3] Design implication: start with a frequent, narrow workflow and make the inference-cost model explicit before promising general intelligence.

Tactical Playbook

Choose a validation wedge by signal, not sunk work. A founder pursuing a broad China-business-operations service has documented roughly 900 teas and 3,500 tea cakes but still has not proved which customer problem should come first. The next-90-day choices are to test repeat paid demand for the tea use case, validate one specific B2B service, or recruit service providers before building more software. [4] Turn that uncertainty into a decision gate: one problem, one leading signal—paid demand, repeat usage, or partner commitment—and one deadline. Catalog completeness is not validation.

Test an AI layer against a live workflow. A website-monitoring builder starts from a solved baseline—detecting page changes—then tests structured extraction such as old/new prices and plain-English alerts for conditions across pricing, products, jobs, documentation, and availability. The builder is asking current users how they work today and offering early access based on useful real-world cases. [5] The transferable sequence is to recruit people with an existing workaround, compare the AI output with their current process, and expand only after repeated usefulness is demonstrated.

Case Studies & Lessons

Agents need behavioral proof, not just successful compilation. Devin's Mac work frames the product problem clearly: an iOS app can compile and launch while still failing to restore a game's state after pause, quit, and relaunch. The agent therefore needs to reproduce the workflow, inspect the running app, change code, and verify the fix. [6] Devin combines accessibility-tree queries for structured controls with screenshots for visual or unrepresented state, then rechecks the result after acting. [6] For agent products, write acceptance criteria as user-visible state transitions; use the cheapest reliable observation channel for each state, and preserve a human takeover path.

Agent-mediated distribution is becoming a concrete platform surface. Muse opened connectors in which developers provide the API while Muse supplies the agent, browser, and context of the user's intent. [7] Meta and Stripe separately announced a partnership to support agentic payments through that platform. [8] Product teams should treat callable actions and commerce—not only the UI—as part of the product surface when users increasingly reach services by asking an agent.

Career Corner

Make judgment legible. A communication framework in the current corpus argues that competence does not automatically become reputation: it is converted through communicating insight, execution, and impact, plus manager advocacy and peer perception. It recommends identifying the company's dominant influence channel and adapting to it; written cultures reward unusually

strong writing. [9] For a high-stakes product review, overprepare, simulate likely questions and objections, lead with impact rather than staffing and schedule detail, then follow up with notes and answers. [9]

Tools & Resources

A practical reference architecture for roadmap/calendar drift. A community-built Sheets add-on treats the sheet as the source of truth, stores each calendar event's ID beside its row so updates do not duplicate events, carries milestone guests in a column, exposes a sync log, and runs automatically. [10] If a team edits plans in a sheet while engineering and marketing operate in Calendar, those are the design choices to inspect; CSV imports, row-level Zaps, and owner-dependent scripts each fail on updates, cost, or maintainability. [10]

Sources

1. substack
2. X post by @ttorres
3. X post by @andrewchen
4. r/startups post by u/itprodavets
5. r/ProductMarketing post by u/Bold_idea_mine
6. X article by @jkelleyrtp
7. X post by @finkd
8. X post by @patrickc
9. How to Communicate to Get Recognized
10. r/ProductManagement post by u/NOOOOOB2