Viljami Kuosmanen

PRODUCT ENGINEER CHECKLIST

How to think like a product engineer

version 2025-08

productengineer.org

Table of Contents

Table of Contents

Product Engineer Manifesto

It is our responsibility as builders to first seek to understand the problem, before diving into solutions.

We concern ourselves with design, technical, and business domains; taking an active part in each to shape the product and empower others to do the same.

Our craft combines Product Thinking with great technical execution.

We take great professional pride in the products we ship and refuse to limit ourselves to technical-only roles in our teams.

The Product Engineer Mindset

In our work as Product Engineers, we have come to value:

- **Continuous delivery of working software** over promises and estimates
- **Asking why** to deeply understand the customer problem before diving into code
- **Customer collaboration and feedback** over tickets and second-hand knowledge
- **Teamwork and communication** over picking up tasks and working in isolation
- **Testing the product ourselves** over relying on others (especially our users) to find issues
- **Domain knowledge and ownership** over outsourcing strategic thinking to someone else

About the Author

Viljami Kuosmanen
Product Engineer

Hi there! I'm Viljami – a software engineer who got tired of being boxed into technical roles and decided to write this checklist to help other engineers step up and start taking more ownership and pride in the products they ship.

In my work as a *Founder*, *Lead Developer* and most recently as *Head of Engineering* leading teams of ambitious senior engineers building multi-million-$ ARR SaaS for the energy industry, along with years of building and maintaining open source products used and contributed to by many great teams and companies, I've learned a set of strategies and questions to help engineers combine technical execution with *product thinking* in order to build products we can be proud to display on our portfolios.

For those interested, this is my product porfolio:

- epilot.cloud (2021–present)
- openapistack.co (2018–present)
- kamerastore.com (2015–2022)
- seravo.com (2013–2017)

What is a Product Engineer?

Not Just Coders, but Builders

Picture this: engineers who don't just speak in code but in product design and customer problems.

Product engineers are a special breed of engineers that don't just see themselves as coders or developers but as builders who deeply care about the products they build.

They're driven by a professional pride and desire to build great products, going beyond the traditional developer role to become genuine drivers who want to put their names behind great products they played a key role in shaping and delivering.

The Counter to Hyperspecialisation

The tech world has seen for a long time a trend towards hyperspecialisation, with engineering roles becoming increasingly narrow and technical.

We have frontend engineers, backend engineers, iOS engineers, DevOps engineers. We have developers defining their careers with a specific language: JavaScript engineers, Swift engineers, C# engineers, Python engineers. We even have developers that seemingly dedicate their entire careers to a single framework or tool: React engineers, .NET engineers, Unity engineers, Ruby on Rails engineers.

Feels like it's only a matter of time until we see job postings seeking for-loop engineers and variable naming engineers.

Jokes aside, specialisation isn't totally without its merits as it has allowed engineers to build deep expertise in specific technologies, a key component in building quality software.

However, it also lead to silos where collaboration and broader product understanding became a nice-to-have and often not even an expectation or focus for engineering roles.

Product engineers stand as a counter-movement to this trend. They embody a holistic approach to engineering, where understanding the entire product and context around it is just as important as the technical skills required to build it. This broad perspective enables them to bridge gaps between different technical domains and ensure that the product serves its customers effectively.

AI: The New Playground

The rise of artificial intelligence (AI) tools in software development is rapidly setting new expectations for engineers. With AI tools becoming more sophisticated and capable, engineers are now expected to leverage these tools to enhance their work, not just in terms of speed and efficiency but also in terms of broadening their area of responsibilities.

Engineers in a post-ChatGPT and Copilot world are now expected to work in more languages, leverage more tools and libraries, and simply deliver more, taking control of a broader range of technologies and disciplines in their daily work.

It's probably not a great time to be the platform engineer whose full-time job is to write scripts and build CI pipelines, or the specialist database engineer who configures databases and optimizes SQL queries, when an LLM tool like ChatGPT can guide any decently smart engineer to do this work without requring deep expertise.

For product engineers, AI tools unlock a higher level of abstraction. With AI making more specialised tasks accessible, engineers can focus their work on creative problem-solving, ideation, and exploring new ways to meet customer needs.

This new level of abstraction allows product engineers to concentrate on product strategy, user experience, and overall system architecture without being bogged down by the intricacies of individual technologies.

The incorporation of AI into software development encourages a broader perspective, where the choice of technology becomes a means to an end, rather than an end in itself.

While AI didn't create the product engineer role – examples have been around for much longer – AI has made becoming a true product engineer much more accessible as a career path for software engineers.

Combining Product Thinking and Technical Execution

Product thinking involves understanding the user's needs, the market demands, and the business goals that drive a product's development. It's about seeing beyond the immediate task to grasp how each piece fits into the broader puzzle of the user experience.

Product engineers don't just build features from specifications. They contribute to the design and roadmap of the product through a robust understanding of the customer's needs and business strategy.

This approach requires a balance of skills: the ability to dive deep into coding and system architecture, while also keeping an eye on the product roadmap and customer feedback.

This checklist contains a set of questions to help engineers hone their skills in exactly these areas. They’re designed to stop you in your tracks before jumping into the solution, and make sure you’ve understood the problem deeply.

I hope this checklist is helpful to those who wish to level up their career to build great products, or break the stereotypes of engineers being thought of as only narrow technologists.

Product Engineer Checklist

1. Understand

1.1 Who’s the user?

As a product engineer, your #1 goal is to create happy users. These are your fans! Always start with the user!

Your user is the person that primarily interacts with your product and whose experience your work will directly impact.

You may have multiple user groups. People who may have varying reasons to use your product or might interact with it in different ways from different angles, maybe on different kinds of devices.

You should understand how to help these users. What drives them to use your product? What delights them? What are their pains?

As a product engineer you should demand and support your team find answers to these questions before jumping into writing code.

1.2 Who's the customer?

Yes, this is a different question to "Who’s the user?". The customer is whoever pays for your product, not always who uses it.

You should know what makes your product valuable to your customers to make better decisions on what to invest your time.

Hint: If you're in B2B the answer always has to do with helping your customers save money or make more money. Understanding your customers' core business is key to understanding why they would pay for your product.

Most importantly, you want to make whoever is paying for your product look good. After all, they're the ones taking a risk by picking your product. You ALWAYS want to reward them for that.

1.3 What's the market?

Zooming out, let's take a look at the wider market landscape. Who are the potential customers we haven't captured yet? Why would a customer pick a competitor's product vs. mine? Are we leaving opportunities on the table?

What can I learn from similar products in the market? What are our USPs? How do I create a competitive advantage against competition?

Are there rules to the market? Are there regulations or industry standards I need to know about? Does my product have to look or feel a certain way to be taken seriously?

Knowing the market and regularly bechmarking yourself against other players helps become aware of your strengths and weaknesses and give ideas for where to invest strategically in your own product.

1.4 Ask Why

This is a bit of a product thinking cliche but still holds true: always pays to ask "why" a few times to uncover root causes of problems and underlying motivations.

Asking why can be helpful in almost any situation to build understanding in your team. Some examples:

- Why should we invest into building this feature?
- Why are customers asking for this feature?
- Why is this a pain for our users?
- Why do users give us that feedback?
- Why now?

1.5 What do we already know?

It's smart to build on what's already known rather than always starting from scratch.

Always leverage the existing knowledge and experience of peers and leaders: founders, product leadership, designers, other product engineers, etc.

Examine the status quo to see how a problem is currently solved. Look at ideas, feedback, metrics, KPIs already collected in the past.

Do we have users already doing something like this? What are their existing workflows? How could we improve their experience?

Am I duplicating or potentially deprecating some functionality that already exists? Could this be achieved by extending or leveraging existing features? How are competitors solving this?

2. Craft

2.1 Am I proud of what I'm building?

Your track record as a product engineer is the products and features you've delivered. Your last feature represents your professional competence level. No excuses.

Ask yourself what quality standard do I want to set for the work I put out there?

Can I be proud of the product I worked on? Is my work well tested and polished? Did I cut corners where I shouldn't have?

As a highly paid professional engineer your craft is to produce high quality software which includes avoiding the creation of technical debt. Never ask for permission to improve quality!

What makes a great product engineer stand out from the average software engineer is an intense sense of professional pride in their work and product.

2.2 Does the product feel good?

This may be a slightly controversial take, but I believe a surprisingly large part of building great products is about developing a good taste for it.

Simply knowing the difference between great vs. average and not settling for just ”ok” helps tremendously in making good decisions as a product engineer.

Does the product feel smooth and consistent? Is it intuitive, simple, and familiar to the user? Or does it feel cheap and janky?

Note that this doesn't only concern the visual aspects of your product. Just slapping a fancy UI design on a shaky foundation doesn't create a great experience.

Simple. Elegant. Clean. This is what we're after as product engineers.

2.3 How do I get there faster?

The pace of innovation especially in software is so rapid that in order to be competitive you must deliver fast, early and often.

Too slow and your customers will lose trust in you while your competitors overtake you.

What many get wrong about agile and building products is optimizing for predictability with estimates and roadmaps. Rather, what you really should care about is visible and continuous progress towards product goals.

The goal of estimates should not be to try to be as accurate as possible but rather to set ambitious and yet achievable goals for yourself.

The question is not how long you think it will take to build, but how long should it take? What's an acceptable amount of time and effort I should invest on this?

What's the rollout strategy? How do I get this into customers' hands as soon as possible? What's the MVP?

2.4 Teamwork

Building products is a team sport.

You are absolutely not expected to work alone and do everything yourself from writing code to doing user research.

Am I effectively communicating with my team? Am I leveraging my team members' strengths? Are we celebrating our successes?

Great teamwork results in great products. Invest in your team, and you will see the dividends in your product's success.

3. Growth

3.1 How do I measure the success of my work?

Our work as product engineers is not just about building and shipping feature after feature.

We make educated guesses about the most valuable thing to work on, so we should also be able to answer the question: What's the impact of my work?

How many customers did we talk to to validate our progress? Are they coming back? How much money does it generate?

Use analytics tools and user feedback to help you understand what's working and what's not. Talk to real users to get qualitative insights about the product.

Most importantly, make sure to share your outcomes openly and transparently. What did I ship in the last few months? What were the results?

3.2 How do I maximise the impact of my work?

The most important question you should regularly ask yourself is: am I focused on the right thing?

Being a good engineer is finding the biggest bottlenecks that hold us back and figuring out how to solve them. You should prioritise your efforts ruthlessly.

Actively communicate what you're working on and why, so that others can help you and keep you accountable. Demo progress frequently and seek feedback.

Don't fear taking risks. Being bold and taking the lead on delivering new and innovative features you believe in can lead to big wins.

Ask yourself: How can I align my work with our company goals? How does my work directly benefit our users? Is there a way I can communicate my work better to others to get better feedback?

3.3 How do I stay ahead of the curve?

Stay updated on market trends. Attend conferences, read up, follow experts.

Make sure to research and benchmark competitor products, or other similar products whenever possible.

Hold frequent retrospectives and brainstorming sessions. Experiment with new ideas. Encourage creativity in your team.

What are the latest trends in my industry? How can I encourage innovation within my team? Are we taking enough time to learn from our successes & failures?

4. Product Vision

4.1 What's our North Star?

To align your work in the context of the broader product vision, you should deeply care about what others in the company are doing and saying, making sure you fully understand and are committed to the overall product strategy. Asking "why" is crucial here.

What are our current product goals? What's our growth strategy? Where do we want to be in the next 2-5 years?

When presenting your work it always helps to put it in the context of how it pushes us forward in the big picture. How does my work help us reach our North Star?

4.2 How does my work impact the design of the product?

Understanding the existing software's design and architecture decisions helps make sure that your work fits well within the larger product design. Always consider how your work influences and is influenced by other components.

When adding new functionality, you should ask:

- Could this be achieved by extending or leveraging existing core features?
- Does my design follow a consistent style to the rest of the product?
- Am I adding complexity or reducing it?

Simplicity is worth fighting for.

4.3 What's the ambition level?

It’s a good idea to define the ambition level when setting off to build something new. Are you aiming for an incremental improvement or a revolutionary new functionality?

Is this feature a USP or a basic expectation? What’s the ROI on building this feature? Does it make sense to spend the effort building this ourselves, or could we use an off the shelf library or service?

Your time is extremely valuable. Think like an owner: Would you invest your salary to build the feature, or rather on something else?

The 3 most important questions

If you've made it this far, kudos to you! But let's be honest. No engineer is going to run through a 15-point checklist to make a product decision. Just seeing the list above can feel overwhelming, especially coming from a mostly technical role.

If we expect engineers to actually start doing this stuff, the core idea of product engineering needs to be simple and easy to remember.

The good news is, to think like a product engineer, it's already enough to start with just three simple questions:

- **What's the problem?**
- **For who?**
- **Why is this important?**

Let's dive a bit deeper into each one.

1. What's the problem?

Stop coding for a second. Do you really know what you're trying to solve? Not the ticket description in Jira, but the real-world issue. If you can't articulate the problem in simple plain English, you have no business writing a single line of code.

2. For who?

Identify your user and customer. Sometimes they're the same person; other times, they're not. Understanding who will use your product (and who will pay for it) is crucial. It shapes the way you approach the solution and helps you tailor the experience to meet their needs.

3. Why is this important?

As engineers there's never a shortage of things we could be improving or adding. If solving the problem doesn't make a meaningful difference, why are you wasting your valuable time? We're not here to build features that nobody uses or cares about. Connect your work to something that actually matters and helps build your track record.

By anchoring your work in these three questions, you immediately move from code monkey to a high value product-minded engineer. Still a rare breed. You're not just implementing features someone else decided to build; you're elevating yourself to a position to influence product decisions and build smarter.

Dealing With Pushback to Product Engineering

I’m sometimes confronted by disappointed product engineers, frustrated with non-engineer colleagues who didn't seem to buy into the idea of involving engineers in the product process from the start.

An all too familiar and frustrating situation for many of us.
You think of yourself as a proud product engineer wanting to solve meaningful problems but then get slapped with a detailed feature specification designed by a group of non-developers without your input. Now they expect you to go deliver their vision.

This is very much the reality in most product teams. Calling yourself a product engineer doesn’t automatically mean your voice will be welcomed. And that’s totally okay.

The ideal vs. the Reality

So far I’ve painted the ideal: engineers who understand customer needs, question roadmaps, challenge designs, and contribute beyond code. But the reality is often much messier.

Some PMs and designers love working closely with engineers. Others are still adjusting to the idea. And that’s fair. The product engineer mindset definitely isn’t the norm.

Let’s not forget, PMs and designers are often under pressure too. It’s only natural they sometimes default to the most familiar and streamlined path. One that doesn’t always include engineers in the early stages.

And let’s be honest. Sometimes engineers haven’t yet built the trust or skills to contribute meaningfully.

The Friction Is Normal

The moment you step outside your lane, you shouldn’t be surprised when the reaction isn’t overwhelmingly supportive.

You will face skepticism.
You might be seen as overstepping.

You will encounter:

- Requests to give estimates for someone else's designs
- Roadmaps shared as top-down mandates
- UI prototypes handed over "ready for dev", expecting pixel-perfect implementation
- Pushback from asking too many questions

This is the price of wanting to do more than just execute. And if we’re honest, many of us haven’t always shown up in these conversations in a way that earns trust.

Is this a culture problem? Not necessarily. Most teams don’t have a rule against engineers joining product discussions. It’s just not the default. It’s less about policy and more about patterns.

Changing those patterns takes trust, initiative, and persistence. At the end of the day, it’s the product engineer’s job to show that product engineering actually works.

Earn the Trust

Calling yourself a product engineer isn't a free pass. No one hands you a seat at the product table just because you want it. You must earn it. *Show, don't tell.*

- Do the homework. Know the names of your customers. Know the business domain.
- Talk to your colleagues, not just other devs. Find opportunities to interact with customers.
- Ask helpful questions that sharpen the team's product thinking.
- Dive into data. Gather insights. Find new metrics and ways to collect useful signals.
- Bring value to the table. Demonstrate you understand the customer problem by making thoughtful proposals that move the product forward.

You need to show up in demos, RFCs, testing sessions, and reviews. Not just in code commits.

Conclusion – Why bother?

Because it's worth it.

We do this for our own professional pride. To put great products into the hands of happy customers. And into our portfolios.

We don't challenge product decisions because we want to take over the PM's job or undermine the work of our design / UX teammates. We do it because we care about impact. Because we want our effort to count.

Because it hurts to pour weeks of your life into something that doesn't work out, and you weren't given a real chance to help make it a success.

Leverage Your Team—They're There for a Reason

The biggest mistake aspiring product engineers make is thinking they have to find all the answers themselves.

Most of us are lucky to work in a multidisciplinary team with non-engineer team members like UX researchers, designers, PMs and business stakeholders whose job is to help answer these questions.

By leaning on your team, you will not only find better answers but also foster a more collaborative and innovative environment.

Product engineering is a team sport.

Our Leverage

And here's something to remember: We, as engineers, hold real leverage. We are the only ones who can actually turn product decisions into reality. No idea ships without us.

So while we may not always get a say by default, we do get a say in how we show up and how deeply we choose to care.

Use that leverage wisely. And proudly.

Start Asking the Right Questions Today

So, the next time you find yourself in a new project or assignment, pause for a moment before diving into code. Ask yourself:

- What's the problem?
- For who?
- Why is this important?

Write down your answers. Reach out to your team for help. This simple practice can transform your approach to work, leading to more impactful decisions and a greater sense of ownership.

Thank you for taking pride in your work as a Product Engineer, making software better for everyone!

Dear Product Manager,

Before anything else: *thank you*.

The work you do, aligning stakeholders, maintaining a relationship with users, finding new opportunities and communicating them is rarely easy and often invisible. It's the kind of effort that can be underestimated from the outside. But we see it. And we respect it.

As product engineers, we're sharing the manifesto not as a critique of your role, but as a challenge to ourselves. We want to step up. Not just write the code, but understand the customer. Not just deliver features, but take responsibility for outcomes.

Still, we know that can come across the wrong way. The things we post: the manifesto, the bold claims, the challenges to not blindly follow roadmaps created by PMs can read like they're aimed at you. But the truth is, we're mostly talking to ourselves. Encouraging each other to break old habits and move beyond being passive implementers.

We want to be useful upstream. We want to sharpen the problem definition. Bring context from past experiments. Understand your goals and constraints. Help shape what we're building before it's locked in. Not because we think we know better—but because we care deeply about getting it right.

However, one of the realities we often see is that having a "manager" title can sometimes create an implicit hierarchy within the team. This can unintentionally lead to a dynamic where engineers are seen primarily as executors of specifications, rather than as creative contributors to the product vision.

The manifesto isn't about diminishing the importance of product management or suggesting that engineers should take over your role. Rather, it's about fostering a space where engineers can engage more deeply with the product vision, contribute ideas, and help bring even more innovative solutions to life. We see this as a way to strengthen our collaboration, not replace it.

So don’t let us off the hook. Treat us as peers. Hold us accountable when we overstep or get lost in abstraction. And when we ask questions “What’s the problem? For who? Why now?” see it for what it is: a desire to think with you, not just follow.

Product engineering is a team sport. We’re here to play it with you.

Sincerely,

Your engineer colleague

Dear Designer,

I hope this letter finds you well. I wanted to take a moment to share some thoughts about how we work together, and more importantly, to express my deep appreciation for the incredible work you do.

Your work is often the soul of the product. The empathy you bring, the deep research and great taste you bring add up to a joyful experience. That's what makes great products feel great. As engineers, we aspire to that level of care.

As product engineers, we often find ourselves in awe of the magic that designers create. Your ability to envision user experiences, understand the needs of our customers, and translate them into beautiful, intuitive designs is truly inspiring. For many of us, designers are like the north star: something to strive towards in terms of creativity, great taste and user empathy.

You might see talk of "product engineering" and wonder if we're trying to step on your toes. And if we're being honest, sometimes our content can come across that way. Manifestos, checklists, strong opinions—written as if we have it all figured out.

But the truth is: it's mostly written for ourselves. It's a reminder to stop coding and start thinking. To look beyond the Jira ticket and ask: Does this feel right? Is it intuitive? Would we be proud to ship this?

We want to be closer to design, not to replace it. Contribute our experience and expertise to it. Because we believe great products happen when engineers develop taste. Not just coding skills.

We're learning. Trying to notice when something feels off. Trying to ask better questions. Trying to be more than just the dev who picks up tickets and needs to be told what to build.

So don't lower your standards for us. Challenge us when we miss the mark. Treat us as partners. Help us grow into engineers who can build with you. Not after you.

Because in the end, product engineering is a team sport. And we want to be teammates you can count on.

Sincerely,

Your engineer colleague