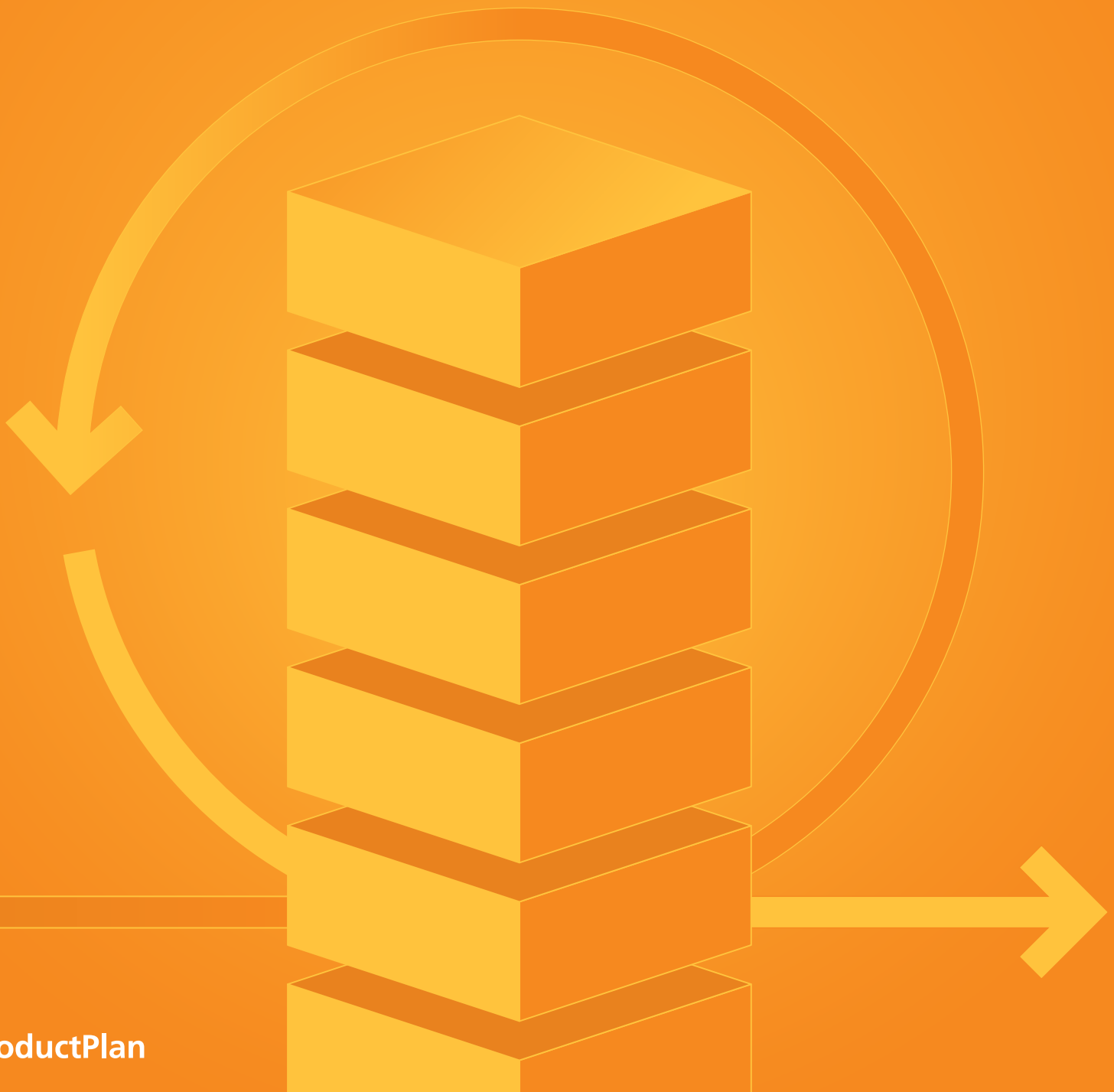


BACKLOG REFINEMENT

How to Prioritize What Matters



BACKLOG REFINEMENT

How to Prioritize What Matters

TABLE OF CONTENTS

Introduction

Backlog Basics

- Warning Signs Your Backlog is Unhealthy

Product Backlog vs. Product Roadmap

Product Roadmap vs. Sprint Roadmap

Backlog Refinement at ProductPlan

- Our Philosophy
- The Product Manager's Role in Backlog Refinement
- 3 Types of Product Managers
- Backlog Ownership
- How Do You Decide What Makes it onto the Backlog?

Our 7 Step Backlog Refinement Process

7 Backlog Prioritization Tips

Key Takeaways



INTRODUCTION

If you've managed a backlog in any capacity, then this scenario might sound familiar:

Your backlog keeps ballooning, with no signs of slowing down. It's a major source of stress for you and the real priorities get lost. At some point, it becomes impossible to accomplish everything you've put in there, especially with how quickly the market changes. What is important now may be irrelevant or invalid by the time you finally get to it.

Throughout my years in product at companies like [AppFolio](#), I've experienced first hand the stress and frustration that managing a backlog creates. At ProductPlan, we've found that ineffective backlogs are often the biggest hindrance to a product manager's ability to successfully drive a product forward.

This is why it's so important to prioritize your product backlog—to make sure it doesn't become an open-ended list of every random thought anyone has about your product. Your backlog needs to be structured, organized, and arranged to favor the most strategically important things for your team to work on.

With a diligent prioritization process and specific backlog criteria, your backlog should not get out of control in the first place.

Ultimately, you are in control of the backlog. A personalized, concrete process will empower you to not only manage the backlog efficiently, but align the entire team on what matters most so you can drive the product strategy forward together.

Jim Semick

Co-Founder

www.productplan.com

BACKLOG **BASICS**



BACKLOG BASICS

Backlog refinement is a recurring event designed to help agile product teams streamline their development process and build better products. It is also referred to as backlog refinement, story time, backlog prioritization, and sizing. Regardless of what your team chooses to call it, the primary purpose is the same—to clarify and prioritize the next few sprints worth of user stories in preparation for sprint planning.

Backlog refinement is meant to be an ongoing mission for you as the product manager. Think of the backlog as your plant; it requires consistent care and nurturing in order to thrive. Without the proper maintenance, the backlog becomes an overwhelming source of stress for you and causes misalignment for your development team.

Warning Signs Your Backlog is Unhealthy

As a product manager, how do you know you've entered the dangerous territory with your backlog? Here are some of the things we've found to watch out for:


- Your backlog has become a dumping ground for every random idea from every stakeholder. Sure, it feels good to be able to tell a vital stakeholder you've "noted" their opinion, but is the minuscule, incremental cognitive overhead worth it if you do that 100 or 1,000 times?
- You're adding ideas that you'd like to implement "some day." This thinking is long-term, and because everything is guaranteed to change from a product, customer, and competitive standpoint, what's the point?
- You're spending hours every month prioritizing items that aren't winding up in your short term sprint backlog.

When you ignore the backlog or maintain it inefficiently, your team loses sight of what matters in the larger context of the overall strategy. You end up distracted by 'shiny objects' and quick wins. User stories become ambiguous and you often miscalculate the amount of time and resources required to complete a task.

Ultimately, you're unable to deliver on what's expected of you.

Not to say that backlog refinement isn't doable, it just requires having the right structure in place. In this book, I'll outline the way we tackle backlog grooming at ProductPlan, revealing our actionable strategies, methodologies, and best practices to prevent your backlog from becoming a black hole.



The background is a solid blue color. On the left side, there are several overlapping, semi-transparent blue rectangular blocks of varying heights and widths, creating a sense of depth. A large, light blue curved arrow starts from the bottom left and points towards the top right, passing behind the main text. Another smaller, darker blue arrow points horizontally from the left towards the center, positioned below the main text.

PRODUCT BACKLOG VS PRODUCT ROADMAP

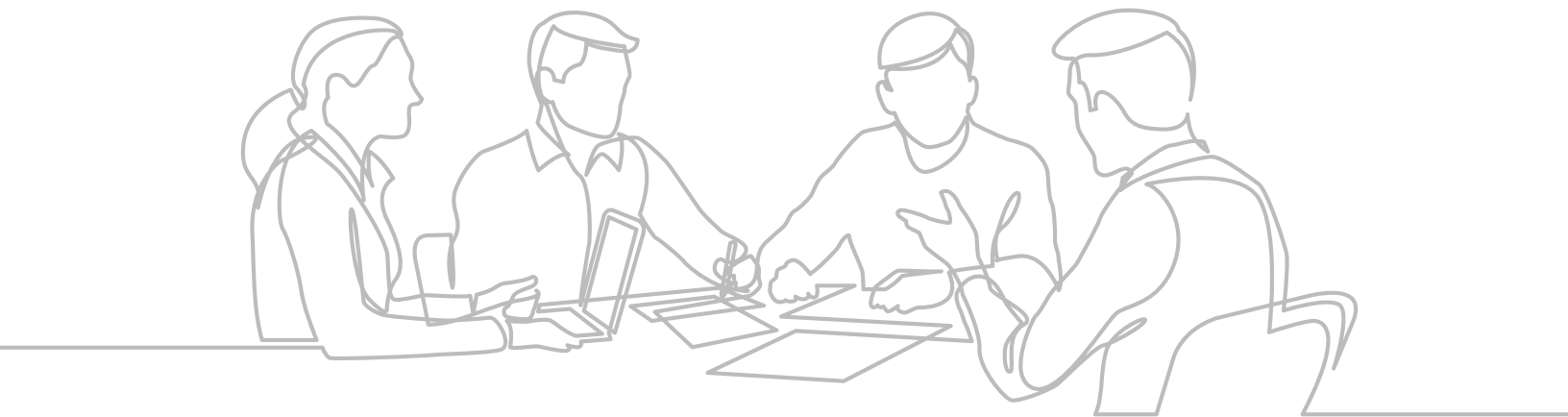


PRODUCT BACKLOG vs PRODUCT ROADMAP

Before we dive into ProductPlan’s backlog refinement process, let’s clarify—the product backlog is not the roadmap. The product roadmap and product backlog serve distinct but complementary roles in helping you shepherd a product from early strategic conceptualization all the way through development and market release. The product roadmap communicates high-level strategic objectives and priorities, whereas the product backlog is a list of tasks that will serve the roadmap’s strategic plan.

There are several key components that differentiate the product roadmap from the product backlog:

Product Roadmap	Product Backlog
Includes high-level themes	Includes task-level jobs such as stories and defects
The audience includes your executive team	An internal document primarily for the product and development teams
Conveys your strategy	Conveys your plan to implement it



Separate but Equal

These tools should be used in conjunction with each other—but not interchangeably.

We discuss the difference in our book, [Feature-Less Roadmaps: Unlock Your Product's Strategic Potential](#).

“Somewhere along the way, the line between backlog and product roadmap blurred. A backlog in the product development context is a prioritized list of items that the team agrees to work on. Typical items on a product backlog include user stories, changes to existing functionality, bug fixes, and features. The features on your backlog are the tactical elements that enable you to deliver your product roadmap.

However, despite our best intentions, features still can find a way of sneaking onto roadmaps—even if they're disguised as a goal or an outcome. Some like the sense of accountability that they provide. Typically, the features appear as task lists, arranged on a timeline (albeit a vague one). Beware of this format. It creates premature commitments and delivery risk.

Features will get a job done, but they shouldn't be the focus at the roadmapping stage. Send them to the backlog. Feature-based roadmaps create external pressure to build the things on the list without ensuring they're solving a real customer problem or asking why the problem is happening in the first place.”

– **Annie Dunham**, Director of Product Management at ProductPlan

Without a high-level overview of your product's strategic objectives and plans, you cannot effectively build a useful list of tasks, in priority order, for developing the product. So you need a standalone, clutter-free, strategic roadmap to capture and communicate this strategy.

At the same time, until you can translate your roadmap's big-picture ideas and strategies into an actionable list of specific tasks—in other words, the backlog—your roadmap won't do much good in helping you drive your product's actual development, because you will not be able to tell your team specifically what to work on next.

So, equally important to your roadmap is an organized and intelligently prioritized product backlog to help keep your development team focused on the right tasks at the right time.

This is also why you do not want to mix these two tools or use only one to serve both purposes. A roadmap with too much tactical detail can cause your team to lose sight of the strategic big picture, and a backlog that focuses on higher-level, strategic items can leave your team without a clear plan for what ground-level tasks to tackle next.



Product Backlog vs. Sprint Backlog

Both the product backlog and sprint backlog are essential to the product planning process, but the two can potentially be confused. For your team to work effectively, you must establish a clear understanding of each backlog's purpose and what items belong to each.

The [product backlog](#) is the comprehensive list of product-related tasks. At any given time, it should encompass all of the things the cross-functional team has agreed to work on eventually, either to bring the product to market or to improve it. Think of it as a working document; one that can be reorganized to reflect shifting priorities.

When these items are kept in order of priority, a product backlog should communicate which user stories, features, bug fixes, and other to-do items the development team should work on next.

You can also think of the product backlog as a tactical, task-level breakdown of the strategic plan outlined in your [product roadmap](#).

Product Roadmap	Sprint Backlog
Comprehensive list of product-related tasks	Shorter list of the most important tasks to complete next
Dynamic and shifts with changing priorities	Static and should remain unchanged
Product roadmap informs the product backlog	Product backlog informs the sprint backlog

With that in mind, a sprint backlog is a much shorter list pulled from the items on the product backlog—specifically, those items the team identifies during a [sprint planning meeting](#) as the most important tasks to complete next.

Since every item is broken down and agreed upon in collaboration with the development team, the sprint backlog should remain static.

Here are a few key takeaways about the distinction between sprint backlogs and product backlogs, and how the two work together:

- 1. Sprint backlog items should be taken directly from the product backlog.**
- 2. While the product backlog can be changed frequently at any time, according to the always-changing realities in an organization or in the market, the sprint backlog should remain as fixed as possible throughout the duration of the sprint.**
- 3. The product team should conduct regular product backlog refinement sessions, to ensure that sprint planning meetings are productive and that the team is able to quickly identify the right tasks to place on the next sprint backlog.**
- 4. The top items on a well-refined, prioritized product backlog will often represent the upcoming sprint backlog.**
- 5. If the team is unable to complete (or even begin) certain sprint backlog items by the end of the sprint, the team might choose to add those unfinished jobs either to the next sprint backlog—if they are still deemed high priority—or to the product backlog to be addressed again in the future.**

The background is a solid orange color. On the left side, there is a series of overlapping, 3D rectangular blocks that create a sense of depth and movement. A large, thick, light-orange arrow curves from the bottom left towards the center right, pointing towards the text. The text is in a bold, white, sans-serif font.

BACKLOG REFINEMENT AT PRODUCTPLAN



BACKLOG REFINEMENT AT PRODUCTPLAN

Our Philosophy

When approaching the backlog, it's important to note that backlog refinement is a personal process. By that, we mean there's no one-size-fits-all; but it should satisfy the needs of you (the product manager), engineering, and UX. Your process will differ depending on your team dynamic, company size, and structure.

To promote alignment among your cross-functional teams, a synchronous, healthy communication flow is your greatest asset. As we communicate with different teams, we have to navigate contrasting personalities, familiarize ourselves with each team's motivations, and understand how we can empower them to find value in the work they're doing. What's important to the engineering team might not align with what UX wants; but it's your role to rally the team around your common purpose—what you're doing and why it matters.

These teams are involved in implementation and execution, so they'll need some of the nitty-gritty details you might leave out in other situations. Don't hold back information because you think it isn't relevant. The more context they have, the better work they'll produce.

They also need an adequate level of detail so they can put themselves in the customer's shoes. They must empathize with the customer to build something that not only works but works for those specific users.

Finally, these folks want to know what they're doing really that matters. Product management represents the customer to the rest of the organization. You're ensuring the teams build products that both satisfy and delight them while advancing the corporate goals and vision.

Regardless of the way you choose to tackle backlog refinement, it is essential that your process facilitates trust and collaboration. Though the backlog is a tactical task, shift your

mindset from viewing it as a list of things that need to get done to a workflow tool that empowers your development team to work together more efficiently.

The Product Manager's Role in Backlog Refinement

Throughout the backlog refinement process, the product manager acts as the facilitator. You want to ensure the conversations remain on track and the right items are being discussed. You're trying to create sanity among the mess of things stakeholders throw at you and ensure that items are being prioritized in accordance with the product strategy. UX might get really obsessed with cool designs and the engineers may dive into the intricacies of code surrounding a given item, but it's the product manager's job to ground the conversation in the perspective that matters most—the customer's.

Above all, you should be able to answer, "Why does my backlog exist?" When you have a firm grasp on the purpose of the backlog and understand what the best use of it is, you can provide the proper strategic guidance.

Our vision for the product is _____. Therefore, the most impactful thing I can do for the product vision is _____.



3 Types of Product Managers

As we mentioned before, your approach to backlog refinement will depend on the dynamics and seniority levels of your team, as well as the problem you're trying to solve.

There are 3 different personas we like to consider:

Pioneers

These product managers are visionary, entrepreneurial, and start products from the ground up. If pioneers have a more senior team, they can operate under a higher level of ambiguity and the team will still be able to function properly. However, if your team requires more detail, the pioneer needs to be careful not to create a dysfunctional relationship if they can't provide a level of granularity that these folks need.

Ideal scenario:

Startup product managers, product managers taking over a new product, product managers with senior-level teams

Settlers

Rather than blazing new trails, these product managers might be refining or slowly iterating a product, or take ownership over a pre-existing product. Either way, the level of detail and understanding in this case is far more granular. This works well for a team that's highly technical and rejects ambiguity. But if you aren't careful, it can cause unnecessary thrash in teams that want more autonomy.

Ideal scenario:

Product managers of mature or pre-existing products or junior-level teams

Hybrid Planners

Planners are a hybrid between pioneers and settlers. They are able to take disparate bodies of information and connect the dots to provide the right detail at the right time. They keep the conversation at the right level, so that the development team understands the high-level 'why', yet doesn't get distracted by the unnecessary technical details.

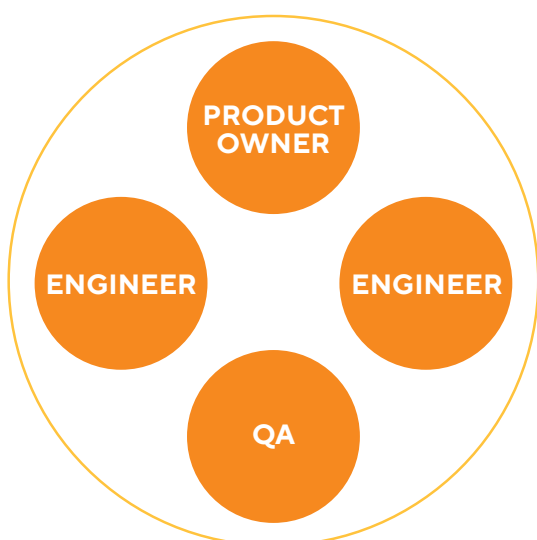
Ideal scenario:

Product managers of mixed-seniority teams

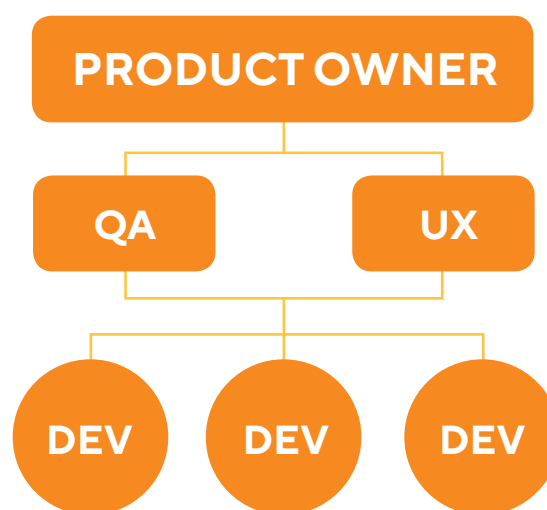
Backlog Ownership

Traditionally, a scrum team would have a product owner, a front end person, a UX designer, engineers, and a QA person. At ProductPlan, our structure is a bit different. We have small teams (2-3) of full stack developers and then a single product owner, QA squad, and UX design team that are shared between the development teams. In some organizations, the product manager—in our case, our Director of Product—plays the role of product owner. They are responsible for the strategy as well as daily participation in scrum team meetings.

Typical Team Structure



ProductPlan Team Structure



Each small team, comprised of developers and designers, maintains autonomy over their own backlog. We store our backlogs in Pivotal Tracker, but you can use any issue tracking system that you currently have in place (Trello, Jira, etc). It outlines or recaps what they will deliver during each sprint. The approach philosophically is that those team members can execute on command without any cross dependencies. With this level of agency, the product manager provides high-level guidance rather than needing to micromanage.

The teams have a commitment to our product manager at a milestone level. It may be delivering a feature, a job to be done, or a user story, and the backlog contains any number of tickets needed to complete that item. It's a working document that everyone involved has access to, but how the teams execute it is totally up to them.

Beyond the individual team backlogs, we have a high-level product backlog, or ‘ideas backlog.’ Essentially, it contains everything that we’d love to do. We store this backlog in Trello to capture ideas from customers and our customer-facing teams. Most people within the organization have access to it and are able to add cards for ideas or feedback, which enables them to feel heard and recognized.

According to our [2020 product management report](#), 39% of respondents are in their backlog at least once per week, with another 32% doing it monthly. Our product manager is in the ideas backlog daily so that they have visibility into what items are being talked about. When items with the same pain point are continuously mentioned, they can identify patterns and levels of importance.



How do you decide what makes it onto the backlog?

To prevent your backlog from becoming a black hole, keep it as lean as possible. We hear this fear often: if an item isn't added to the backlog, then it will be completely forgotten. That mentality creates unnecessary clutter and actually dilutes the items that are truly important. Just as you need to have criteria to help you prioritize what's on your roadmap, you should have criteria that's going to help you prioritize what's in the backlog.

Remember, great products start with "why." Before submitting an item to the backlog, you should ask yourself:

- Why is this ticket here?
- Does this item make a positive impact that contributes to the product's 'why'?
- If we're not doing it now, should it stay on the backlog?

We recommend erring on the conservative side. Spend time to create a philosophy on bugs and enhancements so that you can make a decision as early as possible. Our product team has established a hierarchy for bugs:

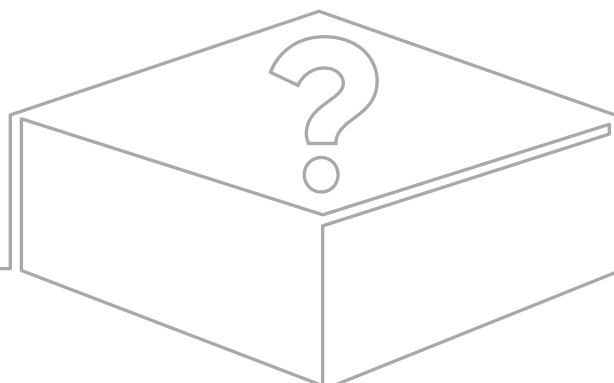
Critical Bugs	Anything that results in data loss and degraded customer experience (they can't accomplish the goal they bought your solution to execute on).
Important Bugs	Anything that handicaps the process or results in your solution not delivering on your perceived value.
Less Critical Bugs	These are the bugs we might not fix right away. This applies to any functionality that works in a different way than intended, but the customer can work around it to get the job done.

Don't be afraid to be selective; every backlog item should meet the criteria of what the product is trying to accomplish. It's easier to deny tickets upfront than have to sort through and delete them retroactively.

The backlog items we have are bugs that are important but not critical, which means that we should fix them in the next sprint or two. However, if you know something should be fixed but you aren't sure where it fits into your timeline or when you will get around to it, don't create a ticket for it. If it is important enough, it will resurface in a later conversation.

This approach isn't for everyone, but our product team removes items that are time stamped and have been in there past a certain date. If we haven't addressed them already after 6 months or they haven't been consistently brought up in conversation, then we don't need to keep them. If this seems too finite, extend the timeline criteria for deleting items to one year.

For some items, it's not as simple as not adding it to the backlog. Sometimes, the question becomes, "where does it belong and at what point has it earned the right to be in the backlog?" The item may belong on your roadmap, your parking lot, or a community forum vote, but it's essential to figure out where in the funnel it belongs before adding it to your backlog.





OUR 7 STEP **BACKLOG REFINEMENT PROCESS**



OUR 7 STEP BACKLOG REFINEMENT PROCESS

1. Long-term Planning

Each quarter, the entire management team holds a priority meeting where we validate what initiatives and high-level goals we think are most important. During this meeting, we have the opportunity to shift priorities around and consider the impact certain items will have on the business. With alignment on the management level, our product team has confirmation on the subset of things we know we want to work on or explore, and the autonomy to execute on them.

2. Backlog Refinement Prep

Everyone who attends a backlog refinement session should do some form of preparation ahead of time. This especially applies to product owners and product managers, whose failure to prepare can lead to frustratingly inefficient sessions.

In our case, our product manager facilitates backlog refinement sessions. It's also not uncommon for the product owner, Scrum Master (in Agile Scrum teams), a project manager, or another team member to lead these sessions. They are responsible for:

- Scheduling the session and inviting the right people to attend.
- Keeping conversations on-topic, focused, and productive.
- Playing timekeeper and moving the conversation forward if the team gets stuck.
- Sending follow up communication to the team after the session.

These events are meant to be collaborative. That means the entire cross-functional team—product owner, QA reps, and delivery team—should be represented at refinement sessions. You need the combined expertise of the various functions on your team to effectively flesh out your user stories.

So before a backlog refinement session, there are a few things we do to get in the right mindset. If necessary, we add any relevant links and steps that we glean from our prep work to the calendar invitation.

1. Revisit Strategic Objectives

Start by taking a step back and reminding yourself of the overarching strategy outlined in the product roadmap. You don't need to overhaul the entire roadmap every time you prepare for backlog refinement, but you do need to keep high-level objectives top of mind.

2. Talk to Stakeholders

Regularly sync up with both internal (executives) and external stakeholders (customers) to get feedback. As a product person, you are a liaison between stakeholders and strategy.

3. Review Key Metrics

If you're running any experiments (which you should be!), you'll want to check in on results. You'll also want to take a look at the health of any key metrics your team is monitoring to take note of any significant changes.

Once you've done a little review, you should feel prepared enough to answer the following questions:

- What are the key themes and stories at the moment?
- Why are these top-priority right now?
- What is their value and how do they match up with our strategic objectives?

With the answers to those questions in mind, you can start looking at what the next few sprints might look like. Most likely, you'll need to re-prioritize the backlog based on new findings and evolving needs. It's wise to shift around priorities before the backlog grooming session rather than during it. However, some teams prefer to re-prioritize user stories in the backlog together so everyone can discuss why things are moving around.

Ideally, you'll have the next two or three sprints worth of stories prioritized before the grooming session. They don't need to be fully developed stories with comprehensive acceptance criteria just yet. Prepare yourself to communicate not only what they are but also why they're important. If you can't explain why something is a high priority in the backlog, it isn't ready to be there.

3. During the Backlog Refinement Session

There is no set time frame for a backlog refinement session. That said, it is not advised to spend excessive amounts of time on these sessions. The general consensus around the ideal length for a backlog grooming session is between 45 minutes to 1 hour.

Efficiency is key with refinement sessions. You need to keep things moving along and ensure conversations stay on track. Some teams decide to assign time limits to each user story to keep things moving. This is where the project manager, scrum master, or another facilitator can be incredibly helpful. Even then, it might seem like you're squeezing a lot of work into a short block of time, but if you're properly prepared you can easily have effective sessions.

If you completed the prep work we mentioned above, then it should be easier to prevent the team from veering off track during the backlog session. Remember to center the conversation around these questions:

- What are the key themes and stories at the moment?
- Why are these top-priority right now?
- What is their value and how do they match up with our strategic objectives?

4. Shaping the Backlog

Inspired by Basecamp, we follow a process they coined as 'Shaping', which is all about taking raw ideas, coming up with a mentally viable solution, and then enabling development teams to wrap their heads around what implementation will be.

There's a standard rubric we use for this process, but the length of time we spend in this phase depends on how complicated the project is. At the end, our desired outcome is to fully understand the scope and the value of each proposed project.

First, the product manager meets with an engineering rep and a UX rep to introduce the problem. When we introduce the issue, we attach an outcome document that explains why this item is important and why now is the right time to do it.

To empower the cross-functional team with the right mindset, share a few relevant customer quotes. It allows everyone to grasp the "why" behind what we're trying to accomplish so that we can all operate from the same baseline. If they understand the root of the problem, the pain that the customer experiences, then the conversations become much easier. There's less push and pull. Otherwise, the outcome can differ dramatically from what we originally intended it to.

Armed with the correct background information, the engineering and UX representatives assess the problem through their lens. We each analyze different things:

- **Engineering:** How can we build code that supports this?
- **UX:** How can we design something that solves this issue?
- **Product:** Where are the rabbit holes in this outcome document?
How long is the business willing to spend on this?

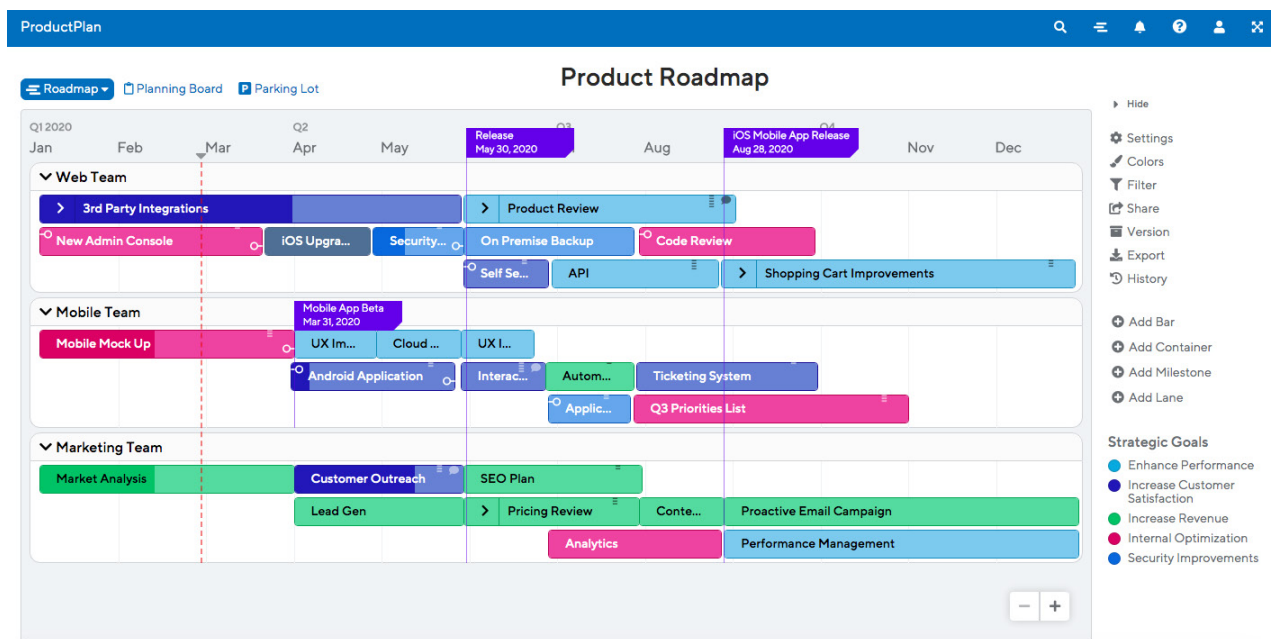
The depth in which we explore any given project will change depending on the estimated time it will take to deliver. If we predict that something will take two weeks, then our perspective and approach shifts.

After we've collected our learnings, we want to have a clear perspective of the scope and value. We want to be able to say for any given project:

In (# of weeks) we can deliver (x solution) and the benefit we think it will have for customers is (x benefit).

5. Betting Table

At this point, everything prioritized on the backlog should drive your product strategy forward and support a high-level strategic objective, or theme. So once an issue goes through the criteria, endures a backlog prioritization meeting, and you decide to pull it onto your product roadmap, there's already a clear connection of where it belongs.



Let's take this product roadmap for example. When you're pulling an item in from the backlog, the item will connect to one of the strategic goals represented on your roadmap. Then, you can easily slot it under the team that will be working on it, and color code it to reflect the goal it's driving.

Once these raw projects come to a state of completion, we put them on what we call a betting table. The betting table is essentially all of these projects that exist as big batch or small batch projects. Big batch being six weeks in duration and small batch being two weeks of duration. Each one of our scrum teams, based on availability, will pick off the betting table. Now we know that there's a team who is going to start on it.

6. Prototyping

Once our team digests the project and understands what they're building and why, they're going to build a prototype. At this point, we haven't submitted any tickets to the backlog yet since we haven't completely tested and made it code ready. The prototype enables us to vet whether or not the solution actually makes sense. The team will present several iterations until we settle on a final prototype.

This process doesn't occur in a silo—we involve our customers as much as possible. Our engineering rep will tackle the code and explore how our solution can support the project while our UX rep is working on the design. As they iterate, they will contact customers to test the prototypes so we can gain insight into how they would actually feel about a certain solution or design. If they don't understand it or see value there, then we know that we need to approach the problem a bit differently.

When a prototype is complete, we write tickets, or more granular tasks, for the backlog. In our organization, each scrum team owns their own backlog. Building the prototype prior to filling the backlog enables us to create these granular tasks with confidence. If we're building or indexing the wrong items, it changes how the backlog is formed. So once they claim a project, the planning session actually enables us to figure out what stories are granular or not in terms of detail to serve those large batch and small batch projects.

Now, it's time to focus on the sprints.

7. Sprint Work

When we focus on the work that will actually be executed during the sprint, we assess the stack of tickets that we created during the prototype phase. The ideal outcome is that everybody has a clear understanding of what the goals are for the sprint. The why and how should be well understood, and there needs to be enough of a granular breakdown to make them successful in terms of what they commit to.

The background is a solid orange color. On the left side, there is a series of overlapping, 3D rectangular blocks that create a sense of depth and movement. A large, thick, light-orange arrow curves from the bottom left towards the right side of the image. The text is positioned in the upper left quadrant, overlapping the blocks and the arrow.

7 BACKLOG **PRIORITIZATION TIPS**



7 BACKLOG PRIORITIZATION TIPS

Ideally, your product backlog should be a list of every product-related task your team needs to complete next, and everything they can and should focus on (within a defined time-frame) after that.

Beyond that point, however, the items on your backlog can quickly become a problem.

This is why it's so important to prioritize your product backlog—to make sure it doesn't become an open-ended list of every random thought anyone has about your product. Your backlog needs to be structured, organized, and arranged to favor the most strategically important things for your team to work on.

1. Determine a bucketing system for organizing items on your backlog.

When organizing your backlog items, it's helpful to set categories that each item will fall under. What you choose to name these categories matters less than the purpose—to give you a clear view of what needs to be prioritized. The backlog feeds what teams will work on during each sprint, so you need a system that empowers you to quickly find exactly what you're looking for.

Some example categories:

Type	Must Do / Should Have / Could Have
Priority	High Priority / Second-level Priority / Low Priority
Size	Large / Medium / Small
Status	Refined / To Be Refined / Not Prioritized

Once you determine the categories your team will use, you can neatly organize and slot in items. Let's take a look at how this would look in practice:

Step 1: Organize backlog items by category

Backlog Example:



High Priority

- _____
- _____

Second-Level Priority

- _____
- _____

Low Priority

- _____
- _____

Bugs

- _____
- _____

Now, when you're planning for your next sprint, you can easily look at the items in the backlog, understand what you need to accomplish, and estimate what your team can handle. This enables us to say:

"Our goal is to deliver _____. To do so, we must deliver these _____ items. We have capacity for _____ amount of work. So we are going to do _____ in order to get there."

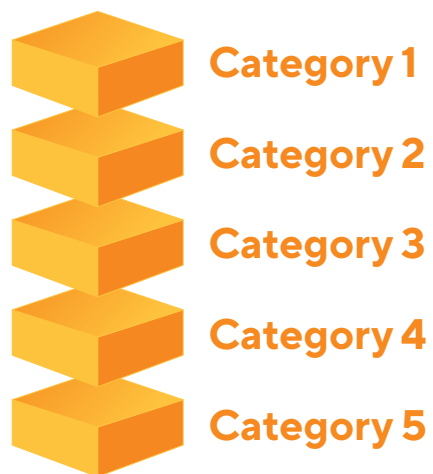
From there, you can cherry pick from your categorized items and pull them into the next sprint. We'll go into further detail on how exactly to score these items and pull them into the sprint backlog below.

Step 2: Pull backlog items into sprint workload

Backlog Example:

Current Sprint

SPRINT 61.0



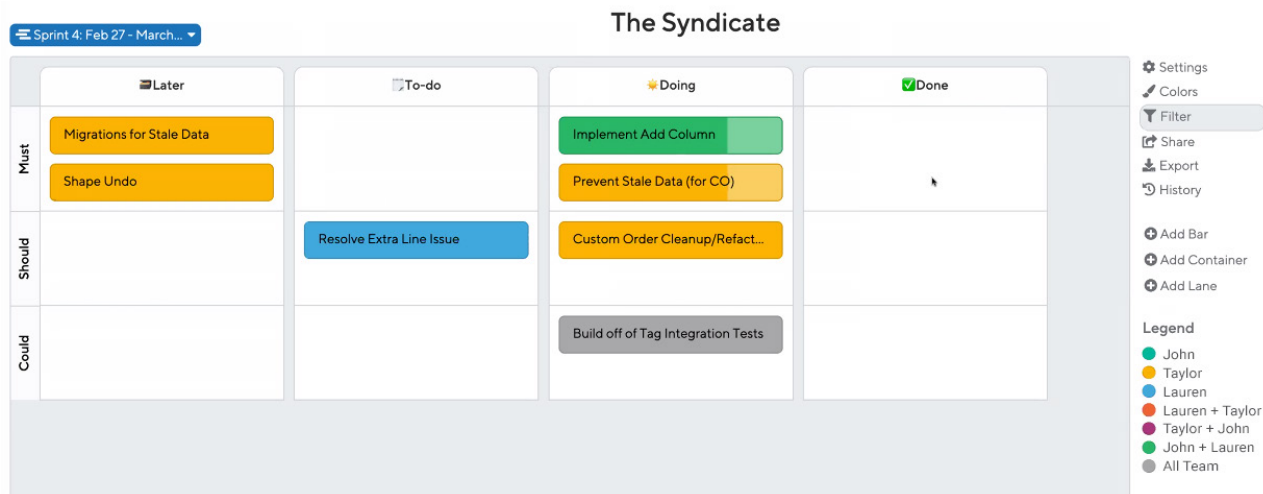
Next Sprint

SPRINT 62.0



Ultimately, your sprint backlog might look something like this:

Step 3: Visualize sprint workload in the tool of your choice



This system provides the structure that your team needs to feel empowered. The more organized your backlog is, the better everyone feels; they know what's coming and can actually move forward.

2. Arrange the top items on your product backlog to represent your next sprint.

One helpful step to organize your product backlog is to arrange the top portion of the list as the contents of your next sprint.

This way you aren't constantly looking at the backlog and asking, "When will we get to this?" and "When can we start tackling that?"

Using this strategy, the top items on your backlog aren't just "top priority" tasks with no internal dates associated with them—they also have a built-in timeline: your next sprint.

Of course, you'll need a mechanism for determining what items should be included in your team's next sprint, and we'll discuss ideas for that below.

3. Don't include any task lower than second-level priority on the backlog.

This is another simple, clean way of determining what makes it onto your backlog and what needs to go somewhere else (like a “Longer-term Tasks” file). Priority level two is a logical cutoff point for what makes it onto your backlog, and here's why.

You've been in brainstorming meetings where the team jots down 20 viable product ideas on the whiteboard. Maybe you've even hosted these meetings. Obviously, you can't execute on all 20 of those ideas, at least not in any near-term timeframe. So what do you do? You prioritize: Maybe you select the best two or four of those ideas and break them into stories, tasks, and plans your team can start working on.

As for everything else on that whiteboard, you'll capture it, of course, but you can't put it all on your backlog (or, even more unrealistically, on your roadmap). The product backlog needs to remain as lean and realistic as possible. It should contain the things on deck for your next sprint, and the second-level priority items you'll get to within the next few months. And that's it.

For tasks below level-two priority, we have another suggestion...

4. Create a separate list for all of those lower-priority (or longer-term) ideas and requests.

What's great about creating a separate list for less-urgent product-related items is that it helps you keep your product backlog limited to those tasks that are truly urgent or of high strategic value. This means it keeps your product backlog more strategically valuable.

Product managers who simply toss every request, idea, and task onto the bottom of their product backlog—because they have no other trusted place to capture and store those items—make every future review and reassessment of their backlog more difficult. They also make it more likely that they will miss something important when they look over their backlog.

So create other lists to capture your product-related ideas that don't earn a spot on the backlog—such as a “Great Ideas” file, and maybe a “Longer-Term Tasks” list.

5. Assign scores (or use some other quantifiable system) for determining each item's overall value.

At ProductPlan, we've included the Planning Board, a [weighted scoring tool](#) in our product roadmap app. We've found that when dealing with a finite amount of time, budget, and development resources, product managers need a mechanism to quantify (or “score”) the overall strategic value of each proposed feature or task against all of the others—to determine which will give their product the biggest strategic advantage. The Planning Board helps you and your team objectively score opportunities to decide what to include on the roadmap.

Title ↓	Lane ↓	Benefit			Cost			Score	Rank ↓
		Increase Revenue	Strategic Value	Customer Value	Dev Effort	Operational Costs	Risk		
	Weight	25	20	10	50	20	20		
Mobile Mock Up	Mobile Team	5	5	5	1	1	1	145	1
UX Improvements	Mobile Team	2	4	5	5	3	1	78	3
Interactive Dialogue Box	Mobile Team	3	1	2	4	2	5	63	6
Accept new forms of payments	Web Team	3	2	3	2	3	1	101	2
New Admin Console	Web Team	1	2	4	2	4	4	77	4
Code Review	Web Team	3	2	4	4	3	3	75	5
API	Web Team	3	2	2	4	4	4	63	6

But you can, and should, take a similar system to score the benefits and costs of items on your product backlog.

We recommend using a scoring model—whether based on ProductPlan's suggested metrics including “Customer Value”, “Increased Revenue” and “Implementation Costs,” or using some other system—to score each item competing for a slot on your backlog.

Some items will earn a spot in your short priority one list (planned for work in the next sprint), others will make it to priority level two (planned for development in, say, the next

three months), and everything else will find itself in your “Longer-Term Tasks” file. But, when you’ve organized your list this way, you’ll know exactly why every item is where it is on your list, and you’ll be able to explain and defend your strategic thinking to your stakeholders and other teams.

6. Figure out a point system for assigning time and development resources to each item.

When prioritizing your backlog, one important factor to keep in mind for every task is how long it will take to complete—and that means not only how many total developer hours but also which specific developers will need to work on the task, and for how long.

Then you might want to convert these hours (or days, or half-days) into points. Hammering out the code for a certain story, for example, might take a full day, which you might want to quantify as one point. This will make it easier to review items on your backlog against each other and calculate needed resources more uniformly across the list (as opposed to saying, “This item should take one developer a half-day, and this one will probably take two developers an hour each.”)

Caution: Remember to keep a task’s “big picture” in mind when trying to estimate how many hours (and whose hours) it will take to complete. For example, you might assume a bug fix is a half-point task—because, as you’ve set up your point system, one point equals one developer day of work. But while it’s true that identifying and correcting the bad code that created the bug might take just a half-day, completing that task will also require writing an automated test for the fix, and actually testing it. So you should be conservative in your time estimates—better to overestimate than underestimate the resources a task will take.

One more warning: Not all points will be interchangeable. It’s important to remember that your team is unique and has a unique set of skills, strengths, and weaknesses. This is why the backlog can play such an important role in your product and development teams’ planning sessions. If you know you have only one or two developers who have the skillset

or experience to handle a certain story or feature, you need to budget the time (the “points”) of those developers carefully as you assign other tasks for your upcoming sprint.

7. Re-evaluate the level one and two items on your backlog regularly.

Finally, it’s important to keep in mind that your product backlog is a living document—changing in priority often. After all, if you’re following the advice in this book, the top portion of your backlog should be disappearing after every sprint, as your team completes them. This means that some portion of the second-level items on the backlog will be moving up after every sprint as well, to the on-deck spot.

If you absolutely must keep a long product backlog because it’s necessary from a corporate or process standpoint, try organizing it or grouping it by a theme, such as “near term” and “long term.” That might help a little bit with your sanity.

As we mentioned previously, you don’t need to track everything that goes into your product backlog. You can use a separate “opportunity” or idea backlog, such as the [Parking Lot](#) in ProductPlan. This approach is a great way to capture ideas that you haven’t committed to, and that needs further validation.

KEY TAKEAWAYS



KEY TAKEAWAYS

1. The product roadmap and backlog should be used in conjunction with one another.

Equally important to your roadmap is an organized and intelligently prioritized product backlog to help keep your development team focused on the right tasks at the right time.

2. Communication is key.

Though the backlog is a tactical task, shift your mindset from viewing it as a list of things that need to get done to a workflow tool that empowers your development team to work together more efficiently. With healthy communication, you'll be able to appeal to each team's motivations to rally them around 'why' behind the work you're doing.

3. Understand how the backlog supports the broader product vision.

Above all, you should be able to answer, "why does my backlog exist?" When you have a firm grasp on the purpose of the backlog and understand what the best use of it is, you can provide the proper strategic guidance.

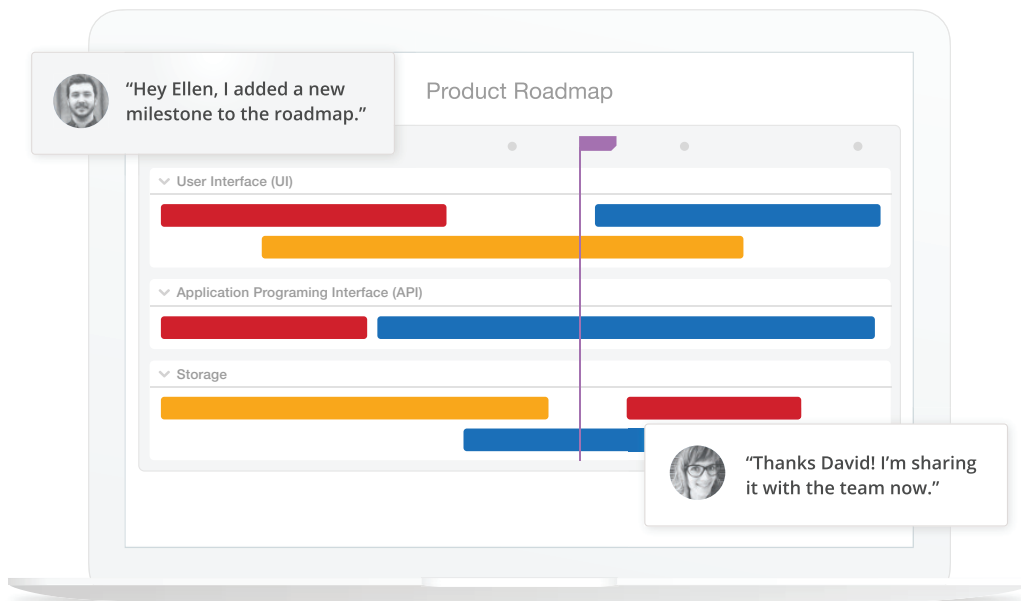
4. Keep it lean.

Overcome the fear that if an item isn't added to the backlog, then it will be completely forgotten. Establish a criteria for backlog items; if it doesn't meet the criteria of what the product is trying to accomplish then it doesn't belong on the backlog.



ABOUT **PRODUCTPLAN**

ProductPlan makes it easy for teams of all sizes to build beautiful roadmaps. Thousands of product managers worldwide—including teams from Nike, Microsoft and Spotify—trust ProductPlan to help them visualize and share their strategies across their entire organization. With our intuitive features, product managers spend less time building roadmaps and more time shipping products.



SIGN IN →