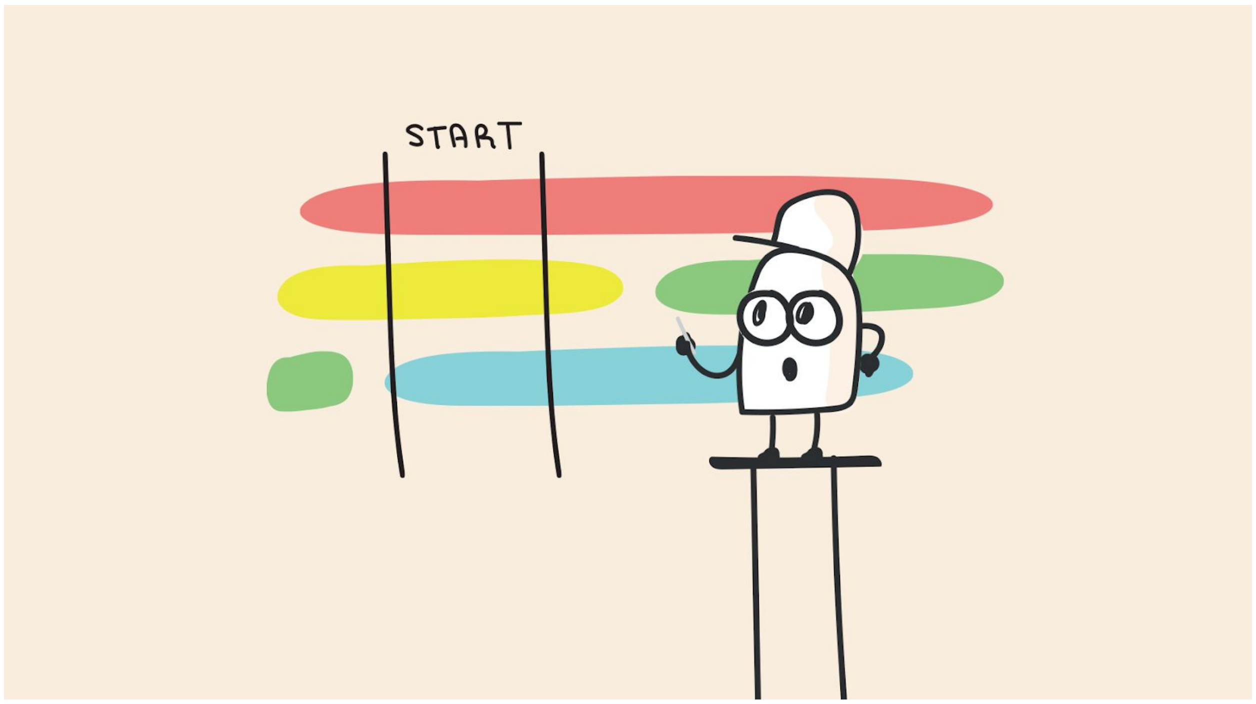
How to Build a Product Backlog that Gets You Results



By Upvoty.com

“I don’t believe in product backlogs.” That’s the discouraging answer we received from the VP of Product at a well-known SaaS company. You see, to write this article, we decided to reach out to different professionals and ask them what makes an efficient product backlog.

And that’s how we received this unexpected answer.

However, instead of getting discouraged, we decided to go all out and identify ultra-actionable insights that will help you create a results-driven product backlog.

Love it or hate it, product backlogs are the core drivers of action. Well, in some cases—because if neglected, your product backlog may become a black hole where you just dump everything without taking a second look.

But let’s take things step by step.

Whether you’re at the beginning of your SaaS endeavor or you’ve already built and worked on several product backlogs, this article has some great tips for all of you. By reading it, you’ll learn:

- The added value of a product backlog and why it’s more than just a list of items and tasks.

- What it’s like to build a second brain for your product team.
- How to avoid building a backlog that will implode.
- How to use the COPE framework to design and maintain a results-driven backlog.
- How to transform your product backlog from a static document into an ongoing process.

We’ve got a lot to cover, so let’s dive right in!

Not your trash can

There are folks in the product community who see “product backlog” as a dirty word. No wonder why: Setting up, maintaining, and using a product backlog requires plenty of energy and time.

Some may mistakenly think that sustaining a backlog is unnecessary and serves more of a distraction than a useful tool. They think that adding, filtering, and prioritizing tasks in a product backlog is a way of procrastination as opposed to the real thing aka building the product.

And that’s true, to some degree.

Think about all the project management tools, systems, and frameworks. There are professionals who’ll spend countless hours on setting up the project, adding tasks, doing the monthly planning, and distributing responsibilities, and then forget all about the document a few days later or, discover that the planning isn’t adopted properly by the team. Obviously, the project management document stops being relevant.

The same is true for product backlogs. You see, a product backlog isn’t a trash bin where you can dump whatever product idea or suggestion you encounter. If you do that, you’ll basically transform your product backlog into a monster that no one dares to approach or work with.

The entire idea of a results-driven product backlog is based on its agility.

Yes, it serves as a document where you can collect all the tasks needed to improve your product, but there’s more to it. The next step is to figure out a way to analyze all the information you’re gathering, and most importantly, retrieve the necessary insights and execute upon them.

In other words, your product backlog is basically the second brain of your team. What’s a second brain, you may be asking?

Keep reading to find out.

Product co-creation and democratization

To answer this question, let’s start with the basics: What’s a product backlog?

No, no, don't stop reading. We won't give you a generic definition we've copied from someone else. We'll share our take on it.

Product backlogs are a tool for co-creation and product democratization.

Products aren't built by a small group of geniuses. They can't survive if they're created in a closed environment, except if you're Elon Musk and building a space rocket. But we're guessing that you're not working on a government project and your product roadmap is public.

By creating and using product backlogs, you're opening your product to your stakeholders, such as teams, investors, and most importantly, your users. You're giving people the opportunity to help enhance your product with their suggestions, comments, feedback, requirements, etc.

In other words, your product backlog is a compilation of data you'll subsequently use to improve your platform while listening to what other people have to say. It carries a stronger meaning than that of a list of changes you need to make. Your product backlog is a way of democratizing your software, opening it to your stakeholders (especially your users), and taking their feedback into consideration for improving the platform.

Understanding the product vision, strategy, roadmap, and backlog

But how are product backlogs different from your product vision, strategy, or roadmap? Some may unfortunately get these mixed up, so let's clarify everything.

👉 Product Vision

In their book *Product Leadership: How Top Product Managers Launch Awesome Products and Build Successful Teams*, authors Richard Banfield, Martin Eriksson, and Nate Walkingshaw explain, "The vision should continue over a very long period of time, potentially years. The best product visions are timeless and disconnected from the technology they are built on." 📝

They also go on to mention, "Disconnecting from time and trends is essential to creating a long-term vision—because, ultimately, achieving the objectives determined by a clear view of the future makes for a better product." In other words, product vision is seen as a long-term framework that aligns the product team with the company's overall goal.

📝 According to Aha!, "A product vision represents the core essence of its product or product line. It also sets the direction for where a product is headed or the end state for what a product will deliver in the future."

In this case, the product vision is seen as the North Star, a guiding tool that helps the team measure their success and make sure they're heading in the right direction. A product vision will usually answer the following questions:

- What's the purpose of our product?
- What problems do we want to solve?
- Whom do we want to impact?
- What impact do we want to have with our product?
- What's the future we envision as a company?

👉 Product Strategy

Product strategy, on the other hand, refers mostly to how you'll achieve the wanted results. People, however, mistake the product strategy for a tactical document.

📝 As author Richard Rumelt argues in his book *Good Strategy Bad Strategy: The Difference and Why It Matters*, "A good strategy has an essential logical structure that I call the kernel. The kernel of a strategy contains three elements: a diagnosis, a guiding policy, and coherent action. The guiding policy specifies the approach to dealing with the obstacles called out in the diagnosis. It is like a signpost, marking the direction forward but not defining the details of the trip. Coherent actions are feasible coordinated policies, resource commitments, and actions designed to carry out the guiding policy."

Simply put, your product strategy should answer the following questions:

- What are the main obstacles and challenges to achieve the overall vision?
- How much we can invest in the development of the product?
- What are the main action lines we'll deploy?

Subsequently, to transform your product strategy into a reality, you need a product roadmap that involves a tactical approach.

👉 Product roadmap

As we've discussed in previous articles, a product roadmap is a framework that includes the steps you'll take to continue developing and improving your product. It's a plan that helps you clearly articulate your product's vision and decide what actions your team must take to execute the strategy.

📝 As ProductPlan highlights, "A product roadmap is a high-level visual summary that maps out the vision and direction of your product offering over time."

Your product roadmap should answer questions such as:

- What features will be built?
- Who’s responsible for building these features?
- When will the new features be released?

According to Jens-Fabian Goetzmann, head of product at 8fit, the product vision outlines the high-level purpose and impact the product should have. It also serves as the “North Star,” which is guiding all product development.

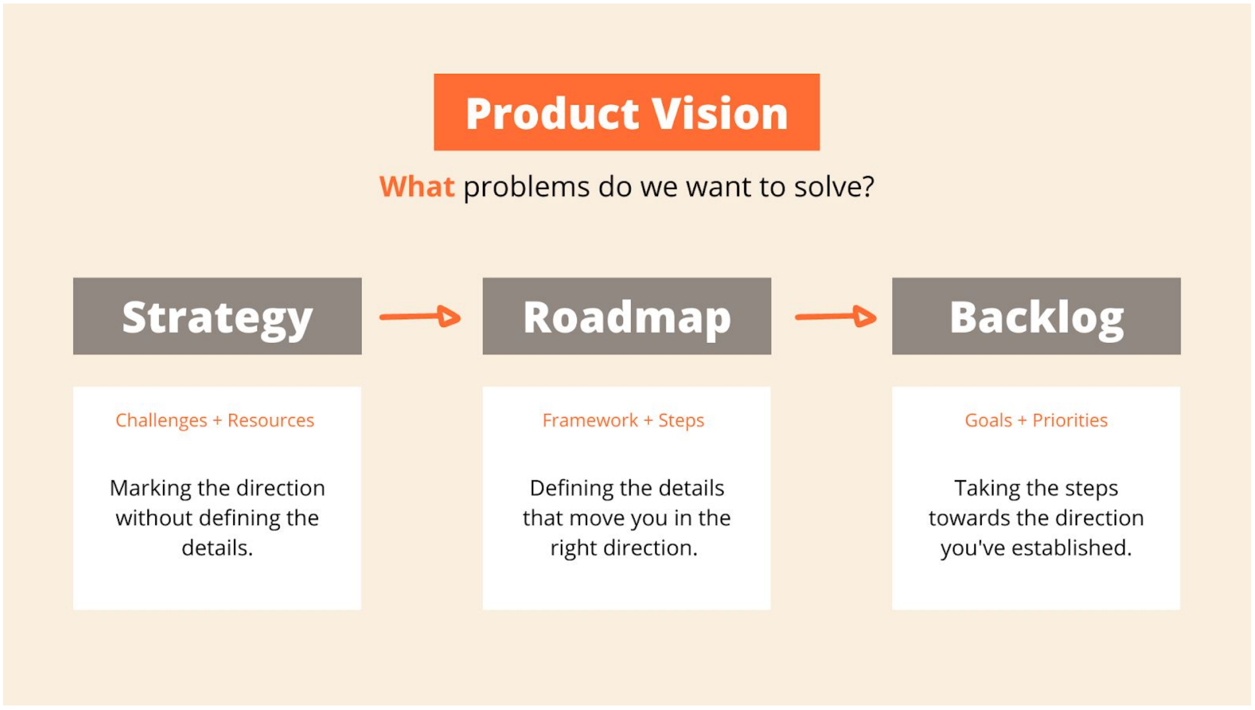
The strategy focuses on user value, competitive position, resources, business model, and growth loops. High-level goals that stem directly from the strategy should be set and checked in a quarterly OKR process.

Finally, the themes in the roadmap must be prioritized based on the strategy but also based on the high-level goals set in the OKR process.

So where does your product backlog fit in this scheme?

As Angus Edwardson, Co-Founder and VP of product at GatherContent told us, **“**Arguably the most important element of a product backlog, would be the context around the product backlog, in terms of how it supports a certain strategy or purpose. This explains the 'why' behind a backlog. An isolated list of features, which don't connect to any greater goal or company objective, is a bit meaningless and not very inspiring, and you don't want your product backlog to be meaningless.”

Your product backlog is a working document where you can capture, filter, and prioritize different ideas, feedback, and requirements with the sole purpose of deciding the next steps in improving the overall product. And that, dear friends, is what a second brain is. Read on—you’ll thank us later for introducing this concept.


Product Vision
What problems do we want to solve?
Strategy
Roadmap
Backlog
Challenges + Resources
Marking the direction without defining the details.
Framework + Steps
Defining the details that move you in the right direction.
Goals + Priorities
Taking the steps towards the direction you've established.

The second brain of your team

Building a second brain sounds fancy, but it's actually one of the most useful approaches you can develop, especially considering the information overload out there these days. Building a second brain refers to creating a centralized digital repository where you can collect valuable information, organize it, and use it systematically to improve different aspects of your life. This becomes absolutely necessary when you have to consume, memorize, and use a great deal of information.

📝 As Tiago Forte, the promoter of the second brain framework indicates, "Being effective in the world today requires managing many different kinds of information – emails, text messages, messaging apps, online articles, books, podcasts, webinars, memos, and many others. All of these kinds of content have value but trying to remember all of it is overwhelming and impractical. By consolidating ideas from these sources, you'll develop a valuable body of work to advance your projects and goals. You'll have an ongoing record of personal discoveries, lessons learned, and **actionable insights for any situation.**"

And although this methodology is used mostly by individuals for personal purposes, you can easily apply it to your product backlog. All you need to do is consider the following attributes a second brain has and extend them to your backlog:

📌 Think like a curator

Let’s digress for a minute and talk about SaaS articles. As you may know, there’s no shortage of blogs, podcasts, and YouTube channels focused on SaaS material. Obviously, it would be impossible to read everything out there. That’s why you’ll probably feel more comfortable following a SaaS knowledge curator who can consume all this information for you and decide what is worth your attention.

The same is true for your product backlog. There are lots of things you can do to improve your product. Likewise, there’s no shortage of feedback from your stakeholders who want to participate in co-creating your platform. However, when creating your product backlog, you’ll want to think more like a curator and pay attention to the existing data, but only include things that are truly aligned with your product vision.

📌 Create a structure

Form and systematization are the key elements that make a second brain truly efficient. The same is valid for your product backlog. You can’t just throw everything in there without considering a specific structure, such as tags, for example. If your product backlog lacks structure, you won’t be able to retrieve the knowledge and use it to improve your product.

📌 Uncover unexpected patterns

The second brain framework focuses on helping you connect the dots. For example, two separate and apparently unrelated inputs can lead to unexpected insight. The same should be true for your product backlog. You may have different user stories that will lead to uncovering an issue pattern that can result in the introduction of a new and extremely valuable feature.

📌 Gain clarity

The entire idea of a second brain revolves around gaining clarity. That’s absolutely crucial, considering the amount of data we have to consume and analyze. Your product backlog should have the same goal and provide you with a clear view. Organizing your backlog for clarity is imperative if you want to be properly equipped with the right mindset and actions to enhance your product.

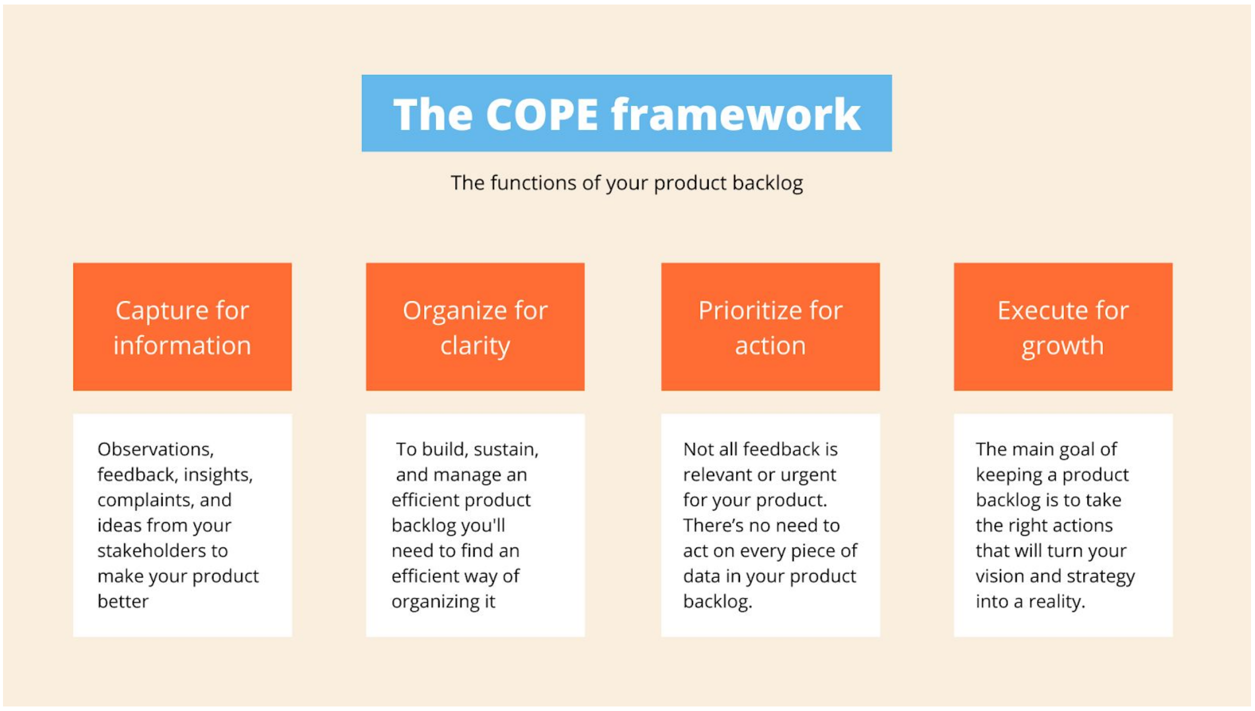
📌 Apply the knowledge into practice

What’s the point of knowledge if you don’t apply it? Storing information, ideas, and facts won’t take you far if you don’t do anything with it. The second brain framework acts like a “magic box” where you can add data, transform it into insights, and then apply those insights to improve something about your personal or professional life.

Results-driven product backlogs have the same attribute. They act as knowledge storage, but with the right filters for organization and prioritization, you can transform the data you’ve added into actions that will improve your product. Having discussed these attributes, let's look at the main functions of a product backlog and how you can take the best advantage of it by using it as the second brain of your team.

The COPE framework

Analyzing how professionals are working on developing their platform or software, we've identified the following functions of a product backlog: capture, organize, prioritize, and execute, or the COPE framework. In the next part of the article, we'll discuss the product backlog through the COPE framework lenses. But first, let's take a look at each individual function.


The COPE framework
The functions of your product backlog
Capture for information
Organize for clarity
Prioritize for action
Execute for growth
Observations, feedback, insights, complaints, and ideas from your stakeholders to make your product better
To build, sustain, and manage an efficient product backlog you'll need to find an efficient way of organizing it
Not all feedback is relevant or urgent for your product. There's no need to act on every piece of data in your product backlog.
The main goal of keeping a product backlog is to take the right actions that will turn your vision and strategy into a reality.

✔️ Capture for information

There's no lack of ideas, data, and feedback when developing a product. Different stakeholders and team members will always have interesting observations that can help you grow and improve your product. That's why capturing data is the first function of a product backlog. And don't forget that you won't be collecting data just for the sake of it—you're capturing it to find actionable insights later and make the right decisions regarding your next steps in product development.

✔️ Organize for clarity

What's a product backlog without a structured framework? It's just a black hole or trash can. To keep your backlog neat and truly efficient, you'll need to be organized.

✔️ Prioritize for action

Not all feedback is relevant or urgent for your product. There's no need to act on every piece of data in your product backlog. That's why prioritization is the third, and probably, most important function.

✔️ Execute for growth

Finally, there's no point in having a product backlog if you don't take any action. The main goal of keeping a product backlog is to take the right actions that will turn your vision and strategy into a reality. And that's possible by applying different execution frameworks your team can use to accomplish tasks in a timely manner.

All these functions will determine whether your backlog is healthy or not. And obviously, you can't expect to work toward your product vision if you have a backlog that keeps ballooning, ready to explode. So let's discuss the differences between a healthy and an unhealthy backlog and how to identify the first signs of inefficiency.

Will your product backlog have a heart attack?

Healthy product backlogs are agile and easy to sustain. They keep ideas moving and help teams take action and develop the product further. Product backlogs that are on the verge of a heart attack, however, are a major source of sleepless nights.

They take too much of your team's energy and attention without giving them the right information to keep up with the market changes. Unhealthy product backlogs are irrelevant and painful to manage.

Here are the warning signs of an inefficient backlog according to ProductPlan:

- Your product backlog has become a dumping ground for random input from your stakeholders.
- Your product backlog is full of ideas you'd like to implement someday.
- You're wasting time and energy on organizing your product backlog without success.

📝 As ProductPlan suggests, "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."

That's a pretty painful experience that will demoralize your team and reduce their confidence in your product. So how do you keep this from happening and maintain a backlog that's agile and results-driven?

To answer these questions, we have to take each function from the COPE framework and discuss how to carry out each one correctly to ensure an ultra-efficient product backlog.

Capture: The quality of your input affects the output

So what items should your product backlog contain?

📝 According to ProductPlan, your backlog is home to a variety of elements, such as new features, bug fixes, changes in existing functionalities, infrastructure updates, and technical debt. But how exactly do these items end up in your product backlog? In other words, who's responsible for the input?

First, this can be you. As a product manager, you likely have plenty of strong ideas on how to develop the platform further. Additionally, this can be your team members who may have different ideas. Next, an important stakeholder who takes part in building your product backlog is the user. You have to pay close attention to what your users say, collect their feedback, and add it to your backlog.

Next, these can be the investors or any other stakeholders that have an impact on your product. But should you accept everyone's input in your backlog? If you do, chances are you'll start growing a black hole. To avoid this, you'll need to pay careful attention to the input and the collection process using the following steps:

👉 Assess the relevance of those providing the input

Let's take your users, for example. Not all customer feedback is the same. Some users are more invested in watching your product grow than others. For example, there's a big difference between the feedback of those people who have been using your product for just a few weeks and those who have been paying customers for six months or more.

Also, there's a big difference between customers who use your software for small tasks and customers who have adopted your platform entirely and are using it to run their entire business. So before adding anything into your product backlog, double-check the source of the feedback. Obviously, you shouldn't discriminate, but at the end of the day, you don't want to add the feedback of your free trial users, except only if it's truly mind-blowing.

👉 Rely on data, instead of your gut feeling

We’re persuaded into believing that every product-related idea should end up in our product backlog. But are all those ideas truly valuable and relevant for your platform’s development?

For example, you may think that adding a new feature is a good idea, although the data shows you otherwise. So instead of going with your gut feeling and mindlessly adding new ideas into the product backlog, when possible, check what the data says. Do your users really need that new feature or improvement? Or should you put it aside and add something more relevant to your backlog?

👉 Weigh the quality of the input

To organize and prioritize your product backlog, you need to assess the relevance and quality of the input before adding it. As we discussed previously, not all ideas are good. To keep your backlog from growing and becoming a black hole, you’ll want to set up a quick quality check system that will help you determine if an idea is worth adding or not.

For example, you could ask yourself if the idea is innovative and can add value to your customers. Or, you could ask yourself if this idea, if made into a reality, will differentiate your product and help keep up with the market. If it does, add it to your product backlog. Weighing input is not an incredibly time-consuming process—it merely takes a few minutes of reflection.

👉 Check if the input is aligned with your product vision

Before adding input to your backlog, always double-check if it’s aligned with your overall product strategy. Similar to the quality check of the inputs, you’ll want to ask yourself if the idea, when made a reality, will help you get closer to the product vision, or if it’s not aligned with it. The answer will help you better evaluate if the input has a place in the product backlog or not.

👉 Create an input capture protocol and make it public

Having all these steps in mind, you’ll want to build an input protocol for the product backlog. This way, you can share it with the team and let people know the evaluation criteria of the items that do end up in the product backlog.

This will be especially helpful, since your team members may have some ideas of their own. You can encourage them to pass their input through the protocol and decide whether their input has its place in the backlog.

As you’ll soon discover, this extra effort during the data input phase will save you lots of headaches during the backlog organization and prioritization.

Organize: Managing the product backlog with less frustration

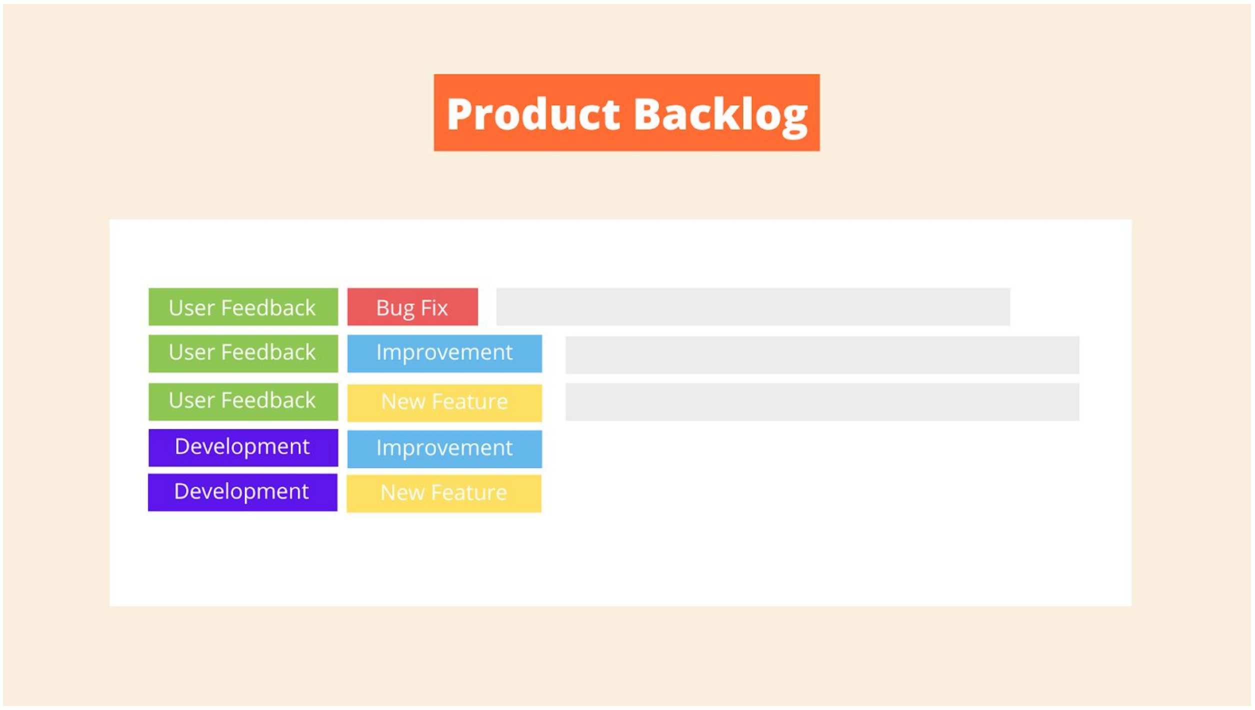
The next step in building and sustaining an efficient product backlog is the way you'll be organizing it. That's crucial, especially considering that there are product managers who feel completely overwhelmed each time they check their backlog. This, obviously, shouldn't happen. That's why you'll want to consider the following recommendations for organizing your product backlog:

👉 Create specific input categories

Do you just throw ideas and feedback into your product backlog? How do you expect to retrieve them if you don't have a way of cataloging all the input? If you do that even for a short period of time, you'll find it difficult to navigate your backlog and soon avoid checking it altogether.

Keep your product backlog organized by creating different categories for your input. These categories can refer to different things and shouldn't be mutually exclusive. For example, you can create a category called "The input source" with variables such as "free trial users," "paying users," "leads and prospects," "product team," "investors," etc.

At the same time, you can create a category called "The input type," and additional variables such as, "bug fixes," "new features," "improvements," etc. Then, you can combine them, and add, let's say, an input labeled "user feedback" of a "bug fix." These categories will help you store the inputs correctly and retrieve them when necessary.



👉 Keep your product backlog clutter-free

Think about your wardrobe. Do you like opening a messy closet where you can't find anything? Or do you prefer a closet that is neat and well-arranged? One of the rules you'll need to follow when organizing your backlog is to keep it simple and clutter-free. Be very careful about the input you're adding, and sort it out from the beginning.

👉 Add input based on projects, instead of tags

Another way you could organize your product backlog, besides creating categories, is to focus on projects. For example, let's say you've developed a marketing automation platform with different solutions, such as email campaigns, chatbots, conversational marketing tools, etc.

Instead of just adding ideas and feedback about different solutions you've created, it's always better to organize your input based on the solutions or projects you'll be developing shortly. For example, you can create a project called "email marketing," and add bug fixes, user feedback, new features, etc. This approach will also help your product team better prioritize your next steps and decide which project deserves attention.

👉 Centralize the management of the backlog

People shouldn't have direct access to your product backlog. Usually, only product managers are responsible for it. So instead of just leaving the backlog open to all sorts of input from different team members, it's better to add a filter to the information and centralize its management.

The last thing you want is to become a bottleneck for growing and maintaining the product backlog. But you can create a separate document and encourage your team members to add their feedback to it, discuss it, and decide what deserves to end up in the product backlog, making sure it stays clutter-free. Then, you can receive these items once per week, review them, and add them to the backlog.

👉 Make a habit out of reviewing your backlog

Last but not least, in some cases, product backlogs will die because their owners didn't get into the habit of reviewing them. But let's not forget that these backlogs require constant attention. At the end of the day, they're your main to-do list in terms of product development.

So you'll want to dedicate an hour once every few days to review the backlog, add new items, check if the organization of the elements is correct, and finally, ensure that your backlog is aligned with your overall product vision.

All these actions will help you keep your product backlog organized and clutter-free. If you want to make life easier for your future self, you'll want to keep your backlog well-maintained. Not only will it help you store information correctly, but it'll also make it possible to easily retrieve and use inputs or user stories.

Prioritize: Knowing what's important

Now that we've dealt with how you can organize the product inputs, let's focus a bit on prioritization. After all, you can add numerous items to your backlog, but you will have to prioritize certain items for execution. There's no shortage of product backlog prioritization criteria, so let's discuss the main ones:

👉 Prioritize based on urgency

What will you give priority: A new feature that will increase the value of your product, or a bug fix that keeps your paying users from accomplishing important tasks? The answer is pretty obvious, isn't it? You want to prioritize your tasks based on urgency. Since you have customers who can't achieve their results because of a bug, you'll want to solve their problem first. After all, you don't want to cause an increase in your churn rate and make people cancel their subscription because of a minor bug that impacts their workflow.

👉 Prioritize based on the value

Another way to prioritize your backlog input is by evaluating the added value these items will provide. You can think both about the market and your users. Will the new feature add value to your product? Will it make it stronger and more desirable? Obviously, you'll want to dedicate more time to those items that will have a greater impact on your business and help you get more customers.

👉 Prioritize based on cost-efficiency

And since we're talking about value, it's also important to evaluate how much it will cost you to accomplish a task on your product backlog. Will you have to spend resources and time on it, while its value isn't that great?

Then again, although a task requires plenty of investments and resources, it might have a big impact in terms of sales. For example, you may want to redesign your entire product to make it more user-friendly.

In this case, you may involve external agencies or have part of your team dedicated to this long-term project. Yet the outcome will have a positive impact on the overall value proposition of your platform, leading to a smaller churn rate and higher revenue.

👉 Prioritize based on relevancy

You can also prioritize your backlog items based on their relevance according to the projects you're currently working on. For example, you may be improving one specific solution inside your platform and some backlog items may be relevant to these changes, and so they become a higher priority.

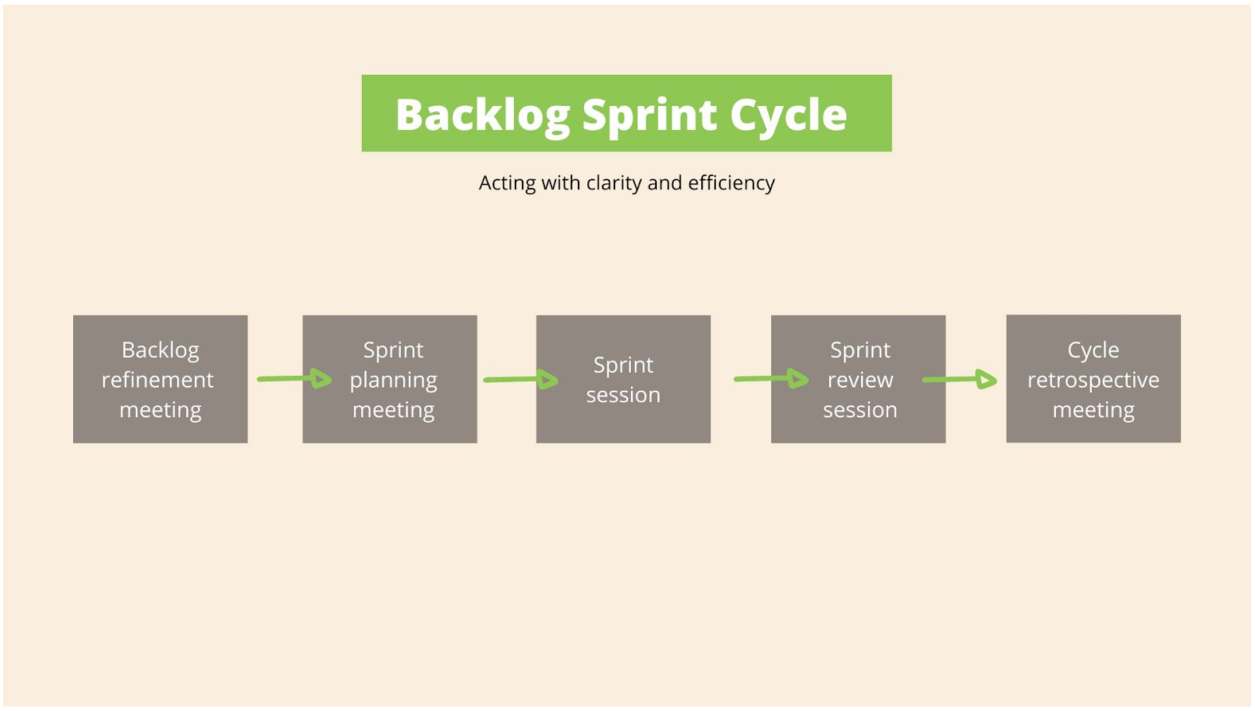
👉 Prioritize based on complexity

You may want to prioritize your backlog elements based on their complexity and the time they'll take to implement. For example, some items may take a sprint, while other times they'll become a slow burn project. In this case, you can develop different strategies.

For example, if you have multiple complex items, you can decide to go one at a time along with implementing easy tasks. But it depends on your approach, your product vision, and the resources you can allocate.

Execute: Acting with clarity and efficiency

Let's talk about action now. How do you take the backlog insights and implement them? There are multiple frameworks product teams can follow, but we're going to focus on scrum and product sprints.


Backlog Sprint Cycle
Acting with clarity and efficiency
Backlog refinement meeting
Sprint planning meeting
Sprint session
Sprint review session
Cycle retrospective meeting

✔️ Step 1: Run a backlog refinement meeting

📝 As Omer Shechter, product manager at Toptal, notes, “After the prioritization process is completed, the next step with the sprint backlog is to create user stories. A product manager inserts initial feature descriptions and includes the raw versions of the user stories into the backlog. Now is the time to engage a scrum team to create new user stories to respond to users’ needs.”

What’s a user story, you may ask?

📝 According to MountainGoat, “User stories are short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. They typically follow a simple template: As a < type of user >, I want < some goal > so that < some reason >.”

The same article points out, “User stories are often written on index cards or sticky notes, stored in a shoebox, and arranged on walls or tables to facilitate planning and discussion. As such, they strongly shift the focus from writing about features to discussing them. These discussions are more important than whatever text is written.”

These user stories can be written during the refinement meeting.

📝 And as ProductPlan notes, “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.”

✔️ Step 2: Organize a sprint planning meeting

Once you’ve prioritized and transformed the backlog items into user stories, you’d better start planning sprints.

📝 According to Atlassian, “A sprint is a short, time-boxed period when a scrum team works to complete a set amount of work. Sprints are at the very heart of scrum and agile methodologies, and getting sprints right will help your agile team ship better software with fewer headaches.”

Additionally, “Sprints make projects more manageable, allow teams to ship high-quality work faster and more frequently, and gives them more flexibility to adapt to change.”

During these planning meetings, the team pulls a small chunk from the top of the priorities list and decides what they need to deliver after the sprint.

✔️ Step 3: Run a sprint session

During a short time frame (usually two to four weeks), your team needs to design, code, and test the pieces you've decided to implement. Throughout the entire sprint, the team needs to check in daily to share their progress or any potential challenges that may impact the deadline. At the end of the sprint, they need to present the deliverables.

There's an important thing to remember, though. According to Angus Edwardson, Co-Founder and VP of product at GatherContent, "How do you know if the work you have delivered has had an impact? A lot of product teams simply define success as moving something into 'done'. This does not tell you whether or not the thing you have built truly works for the people that use it, or if it is having its desired impact - delivering value. With that in mind, it's important to have a clear definition of success. This can be in the form of a binary goal for a feature or piece of work, or maybe a metric you're trying to move, or even just some behaviours you are monitoring. This encourages follow-through, and an iterative flow of improvements."

✔️ Step 4: Set up a sprint review session

📝 As Dan Radigan, technical account manager at Atlassian, highlights, "Sprint reviews are not retrospectives. A sprint review is about demonstrating the hard work of the entire team: designers, developers, and the product owner."

During these sessions, your team members and stakeholders should be reviewing the deliverables, asking questions, and giving feedback. Also, don't forget that these sessions are a great opportunity to celebrate your team and thank them for their accomplishments.

✔️ Step 5: Round up the cycle with a retrospective meeting

During these sessions, the team can discuss any efficiency or behavior-related details they need to adjust or improve.

📝 According to Visual Paradigm, "This is at most a three-hour meeting for one-month Sprints. The retrospective session is an 'improvement' meeting held to find ways and means to identify potential pitfalls, past mistakes, and seek out new ways to avoid those mistakes."

During this meeting, you'll want to answer questions such as:

- What went well during the sprint?
- What went wrong during the sprint?
- What did we learn?
- What can we improve?

Then, take any corrective measurements necessary and repeat with a new sprint planning meeting.

Your Product Backlog: An ongoing process

Product backlogs can be scary, but only when we're seeing them as static documents that you need to feed with constant input. To transform these backlogs from black holes to results-driven tools, you'll want to change your perspective.

You see, whenever you think about a product backlog, you're envisioning a repository. But what if you started to view your product backlog as an ongoing process that needs to be incorporated into your daily workflow?

And that's something you can easily do by remembering a few quick things.

Quick recap

- Your product backlog is not just a list of ideas and tasks to further develop your platform. It's also a way to expand access to your product offerings and provide more opportunities for growth through stakeholder feedback.
- Your product backlog is the second brain of your team. It allows you to think like a curator, create a structure around feedback, uncover unexpected patterns, gain clarity, and transform insights into actionable steps.
- There's a big difference between black hole backlogs and healthy backlogs. The first one is basically a dumping ground nobody cares about, while the second one will contribute to the growth of your product.
- Using the COPE framework will help you build and maintain a results-driven product backlog. As a reminder, COPE stands for capture for insights, organize for clarity, prioritize for action, and execute from growth.
- Always remember that the quality of your input will affect the output. So make sure you're assessing who's providing you with the inputs, weigh the quality of the items you'll be putting on the backlog, check that the input is aligned with the product vision, and create and input protocol to ensure transparency.
- To make things easier for your future self, always organize your product backlog by creating categories and tags for your items, and then add the items to different product projects you have open. Keep your backlog clutter-free by centralizing its management and adding an extra filter for input evaluation.
- Next, take your product backlog through item prioritization by assessing whether an item is urgent or relevant to your users. Also, evaluate the cost-efficiency and the value each item can add to your overall product.
- Finally, you need to execute your product backlog. Run a backlog refinement session during which you can transform items into user stories. Then, set up sprint planning meetings to agree upon the next tasks that will be implemented. Subsequently, the team will start to design, code, and test the new improvements during a short period of time. The deliverables will be presented during the sprint review session. You'll round up

everything with a retrospective meeting to discuss the necessary adjustments. Then, you'll be ready to repeat it all over again.

Final thoughts

The key thing to remember is that your product backlog is not a static document; rather, i**t's a process that integrates perfectly into your business life cycle**, from marketing your product and getting new customers to collecting the feedback of your users, integrating it into your product backlog, and working upon it, delivering the expected features or fixing the annoying bugs.

It's an ongoing process you'll want to incorporate into the workflow of your teams to make sure that everyone is in alignment. Otherwise, you'll just see the product backlog as an annoying concept that doesn't bring any value. So instead of having a skeptical view about it, you may want to give it a try and streamline the constant growth and improvement of your product.

By Upvoty.com