Feature Prioritization

Every SaaS company might face a common challenge: **too many great ideas and limited resources to implement them**. Whether you're just starting out or managing an established product, you'll receive feature requests and suggestions from everywhere: your customers want new capabilities, your support staff identifies pain points or your competitors launch new compelling features. But you can't build everything at once.

In this and the next two lectures, we'll explore how successful SaaS companies make smart decisions.

1. Identify features that bring the most value → **Feature Prioritization**
2. Organize and track prioritized features → **Backlog Management**
3. Turn prioritized features into actionable plans → **Product Roadmap**

We'll begin by exploring **proven methods for feature prioritization** that help you identify which additions will bring the most value to your business. Then, we'll look at practical approaches to organizing and tracking these prioritized features through effective **backlog management**. Finally, we'll cover how to transform your prioritized features into an actionable **product roadmap** that guides your team's execution.

First, **we need to know what to build**. In other words, we need to **figure out which features will bring the most value to our users and make the biggest impact**. To help with this, there are many methods for feature prioritization. Now, we're going to discuss widely used approaches that can guide these important decisions.

Value vs. Effort Matrix

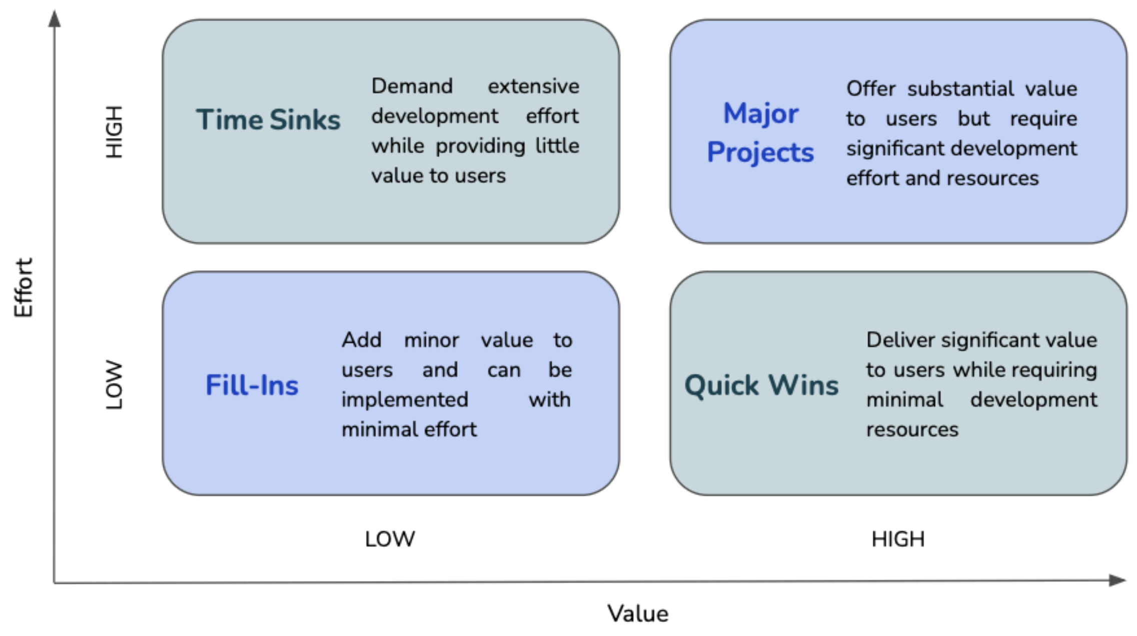
The first practical framework that we will discuss is called: **Value vs. Effort Matrix**. This approach helps the teams make quick yet informed decisions by evaluating two key factors:

- **the potential value a feature brings to users**

- **and the effort required to implement it**

The framework can be visualized as a grid divided into 4 quadrants. On the horizontal axis, you measure **the value a feature could deliver to your users and business**. On the vertical axis, **you assess the effort** needed to build that feature, including time, resources, and complexity.

This creates four distinct categories that help guide your decision-making:


Effort
HIGH
LOW
Time Sinks
Demand extensive development effort while providing little value to users
Major Projects
Offer substantial value to users but require significant development effort and resources
Fill-Ins
Add minor value to users and can be implemented with minimal effort
Quick Wins
Deliver significant value to users while requiring minimal development resources
LOW
HIGH
Value

- ***Quick Wins (high value, low effort):***

Quick Wins are features that are your priority targets. They **deliver significant value to users while requiring minimal development resources**. These are the features that can give you the most value for your investment and should be at the top of your development list.

- ***Major Projects (high value, high effort):***

Then we have Major Projects. These are **high-value features that could significantly boost your product's success but require substantial resources to implement**. While they offer strong potential, they need careful planning and might need to be broken down into smaller, more manageable pieces.

- ***Fill-Ins (low value, low effort):***

Next, Fill-Ins are **nice-to-have features that don't require much effort to implement.** While they shouldn't be your top priority, they can be good candidates for times when your development team has extra capacity or when you want to show continuous product improvement.

- ***Time Sinks (low value, high effort):***

Finally, we have a category called Time Sinks. These features **should generally be avoided or reconsidered**. They **require substantial resources but don't deliver proportional value** to your users or business. Unless there's a compelling strategic reason, these features should be at the bottom of your priority list.

Let's look at a practical example of developing a team collaboration tool. We're thinking about several potential features to add, and we'll use a Value vs. Effort Matrix to evaluate each one.

- *Adding keyboard shortcuts*
- *Building an AI-powered document scanner*
- *Changing button colors*
- *Creating a custom video editor*

First, we want to add keyboard shortcuts. This feature would significantly speed up how users interact with our platform, making their daily work more efficient. Since implementing shortcuts requires minimal development effort but can dramatically improve user experience, it's an obvious quick win.

Secondly, we are considering an AI-powered document scanner. While this feature could greatly help teams handle document processing and potentially attract new enterprise clients, it requires substantial development resources and AI expertise. Therefore we put this feature in the quadrant Major Projects.

Changing button colors can be placed in the Fill-Ins quadrant. While this feature might bring minor aesthetic improvements to our interface, it won't significantly impact how users work with the tool. The development effort would be minimal and it could be implemented alongside more important features if resources allow.

Finally, we have the custom video editor feature. This is a classic example of a feature that might seem attractive but doesn't belong in a product's core functionality. The development effort would be massive - requiring months of development time, specialized expertise in video processing, and ongoing maintenance. At the same time, the value to users would be minimal since video editing isn't a primary need for most teams using a collaboration platform. They're more likely to use dedicated video editing software when needed. This makes it a clear Time Sink.

RICE Scoring Framework

Let’s move to the second method called the RICE scoring framework. It helps to prioritize features by **combining key factors into a single score**. You can see the formula used to calculate the RICE score on the slide.

$$\text{RICE (Reach, Impact, Confidence, Effort) scoring framework} = \frac{\text{(Reach} \times \text{Impact} \times \text{Confidence)}}{\text{Effort}}$$

We multiply the scores for **Reach**, **Impact**, and **Confidence**, and then divide the result by the **Effort** score. Now, let’s break down each of these factors in more detail.

REACH

First, Reach score. Think of Reach as **the number of users a feature will affect in a specific time period**. Typically, it's measured quarterly, but you can also choose other periods, like monthly or biannually. Although you can’t know exact future usage before building the feature, you can make informed estimates. If you have access to real data, it can make your predictions more accurate. By analyzing key data points, you can base estimates on actual trends rather than assumptions.

- *Usage data from similar existing features*
- *Customer feedback and feature requests from users*
- *Survey responses where users indicated interest in such functionality*
- *Market research and user behavior patterns*

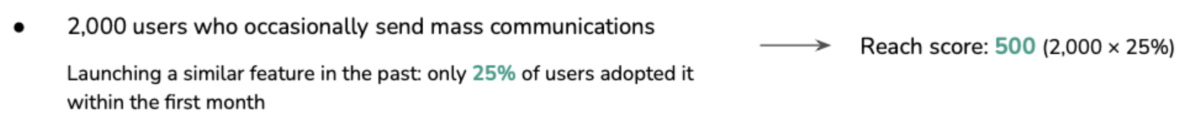
For example, you can **analyze usage patterns of similar features** you've already launched to predict adoption rates. **Customer feedback and feature requests** give you direct signals of interest - if many users are asking for something, they're likely to use it. **User surveys** where people express interest in specific functionality can also indicate potential reach. And **market research and behavioral data** help you understand broader patterns of user needs.

Let's say we're considering adding a bulk email feature to the platform.

- 400 of current users regularly send over 1,000 emails per month $\longrightarrow$ Reach score: 400

Our analysis shows that 400 of current users regularly send over 1,000 emails per month. So, our Reach score is 400.

However, the calculation might be more complex, so let's consider a different scenario for the bulk email feature.



- 2,000 users who occasionally send mass communications

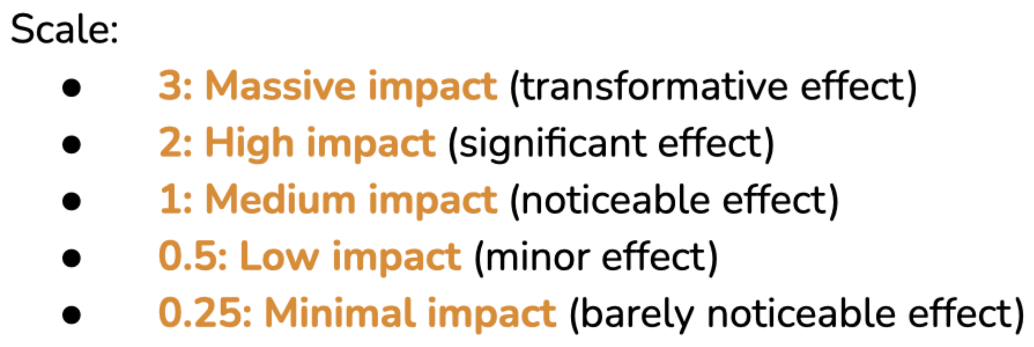
  Launching a similar feature in the past: only 25% of users adopted it within the first month

$\longrightarrow$ Reach score: 500 (2,000 × 25%)

The data shows we have 2,000 users who occasionally send mass communications. From past experience, when we launched a similar feature, only 25% of users adopted it within the first month. We can use this figure to predict the adoption rate for the bulk email feature. So, we rely on historical data to make an informed estimate about future behavior of users. In this case, the Reach score would be 500.

IMPACT

The next component is Impact. It measures **how strongly a feature will affect your users and business goals**. When considering a feature's impact, we assign it a score on a scale.



Scale:

- 3: Massive impact (transformative effect)
- 2: High impact (significant effect)
- 1: Medium impact (noticeable effect)
- 0.5: Low impact (minor effect)
- 0.25: Minimal impact (barely noticeable effect)

A score of **3** points represents a massive impact. For instance, the feature that transforms the core user experience. Next, features that significantly improve user workflow earn 2 points. When a feature creates a noticeable improvement, it receives 1 point for medium impact. Next, minor enhancements that don't dramatically change the user experience get 0.5 points. And finally, changes that users might barely notice receive 0.25 points for minimal impact.

Let's get back to our example of adding a bulk email feature. When scoring impact, we need to evaluate how the feature will affect our business goals and user needs. First, we look at how it supports business objectives by asking key questions.

Assessing how it affects business objectives:

Will this feature help convert more users into paying customers?

Will it boost user retention?

So **will this feature help convert more users into paying customers?** Well, we assume that limited email functionality can be a potential barrier for some trial users considering an upgrade. Adding this feature might address that limitation and help improve conversion rates.

Secondly, **will it boost user retention?** Our customer support team frequently receives inquiries about bulk email functionality. Market research also shows that competitors already offer this capability. So, if we provide this feature, it could reduce the likelihood of users switching to other platforms.

By answering such questions, we can assess how well the feature aligns with business goals and whether it deserves a high Impact score.

Next, we assess how well the feature addresses user needs.

Assessing how it affects user needs:

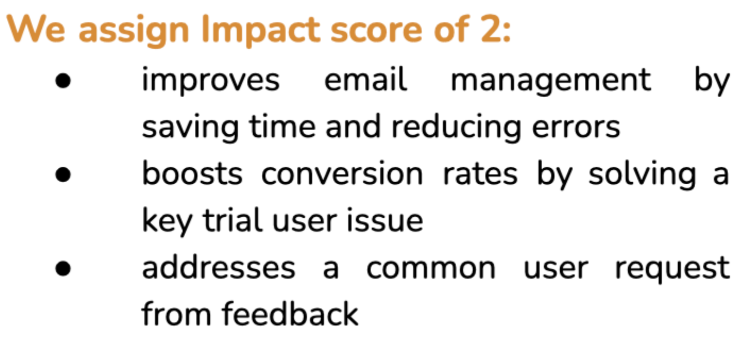
Will it save users time or make their work more efficient?

Will it improve accuracy and reduce mistakes?

We can ask ourselves **whether the bulk email feature could save users time**. Our research shows that users currently spend hours sending emails one at a time. By allowing them to send hundreds of emails in a single action, we could streamline their workflow and provide significant value.

We also know that manual email campaigns are prone to errors, such as missed recipients or formatting issues. Automating this process would likely minimize these mistakes and thus enhance the overall user experience.

Based on our analysis, we might assign the bulk email feature an Impact score of 2.



We assign Impact score of 2:

- improves email management by saving time and reducing errors
- boosts conversion rates by solving a key trial user issue
- addresses a common user request from feedback

This is because it has a significant effect on how users manage their email campaigns by saving time and improving accuracy. Also, it has potential to boost conversion rates by addressing a common limitation that prevents trial users from upgrading. And finally, it directly addresses a frequently mentioned user need, as seen in support inquiries and feedback.

CONFIDENCE

Now, let's move to the third component in our formula which is Confidence. It represents **how sure you are about your Reach and Impact estimates**. Even with data and careful analysis, some level of uncertainty always exists when predicting how features will perform. That's why we include Confidence as a reality check in our scoring process.

We express Confidence as a percentage, using three basic levels.

Levels:

- **100% - High confidence:** We have reliable data supporting both Reach and Impact estimates.

- **80% - Medium confidence:** We have solid data for one aspect but rely more on educated guesses for the other.

- **50% - Low confidence:** Estimates are based primarily on assumptions or indirect evidence, rather than hard data.

- ***100% - High confidence***

First, we can assign 100% confidence when we have **reliable data supporting both Reach and Impact estimates.** For example, we might have clear usage statistics and direct feedback from users to back up our estimates.

- ***80% - Medium confidence***

We can consider using 80% confidence in cases where **we have solid data for one aspect** (either Reach or Impact) **but rely more on educated guesses for the other**. For instance, we have clear usage data about the reach but less concrete evidence about potential impact.

- ***50% - Low confidence***

And 50% confidence level applies when our estimates are based **primarily on assumptions or indirect evidence**, rather than hard data. If we are less than 50% confident, we probably need to do more research before evaluating the feature.

In general, if we are excited about a feature but don't have much data to support estimates, **a lower confidence score will adjust the overall RICE score to reflect this uncertainty.**

- ***Reach**: We have solid usage data and historical trends that provide strong evidence for predicting how many users would adopt the feature.*
- ***Impact**: We have competitor analysis and user feedback to support its importance, but some effects, like improvements in conversion rates, rely on assumptions.*

When evaluating Confidence for the bulk email feature, we consider the reliability of the data supporting our estimates. For Reach, we rely on solid usage data and historical trends. These sources provide strong evidence for predicting how many users would adopt the feature. For Impact, our evaluation is less certain in some areas. While competitor analysis and user feedback highlight the importance of this feature, other aspects, such as its exact effect on conversion rates, are based on our assumptions rather than direct evidence.

As a result, we assign a Confidence score of **80%** because Reach is supported by strong data, while Impact relies partly on assumptions.

EFFORT

The final component is Effort, which serves as the denominator in the RICE formula. Effort **measures the amount of work required to implement a feature**. It is expressed in terms of **person-months**. **One person-month represents the work one team member can complete in a month**. This includes not just development time, but also design, testing, and any other resources needed to develop the feature.

By dividing the combined value of Reach, Impact, and Confidence by Effort, we ensure that features delivering the most value relative to their cost are prioritized.

For our bulk email feature, we need to consider several key development areas. The most substantial work involves building the backend system to manage email queues and handle the sending infrastructure. The frontend requires creating an intuitive interface where users can manage their bulk email campaigns and control sending parameters. We'll also need to integrate this feature with existing email service providers to ensure reliable delivery. Also, security and performance are critical considerations. We must implement robust security measures and conduct thorough testing to handle large email volumes effectively.

- ***Building Backend system*** *to manage email queues*
- ***Creating an intuitive interface*** *for managing campaigns and sending parameters*
- ***Integrating the feature*** *with existing email service providers*
- ***Implementing robust security measures*** *to protect data*
- ***Conducting thorough testing*** *to handle large email volumes effectively.*

Taking all these factors into account, we estimate the total effort at 960 hours or approximately 6 person-months of work. In other words, the total effort required is equivalent to 1 person working full-time for 6 months.

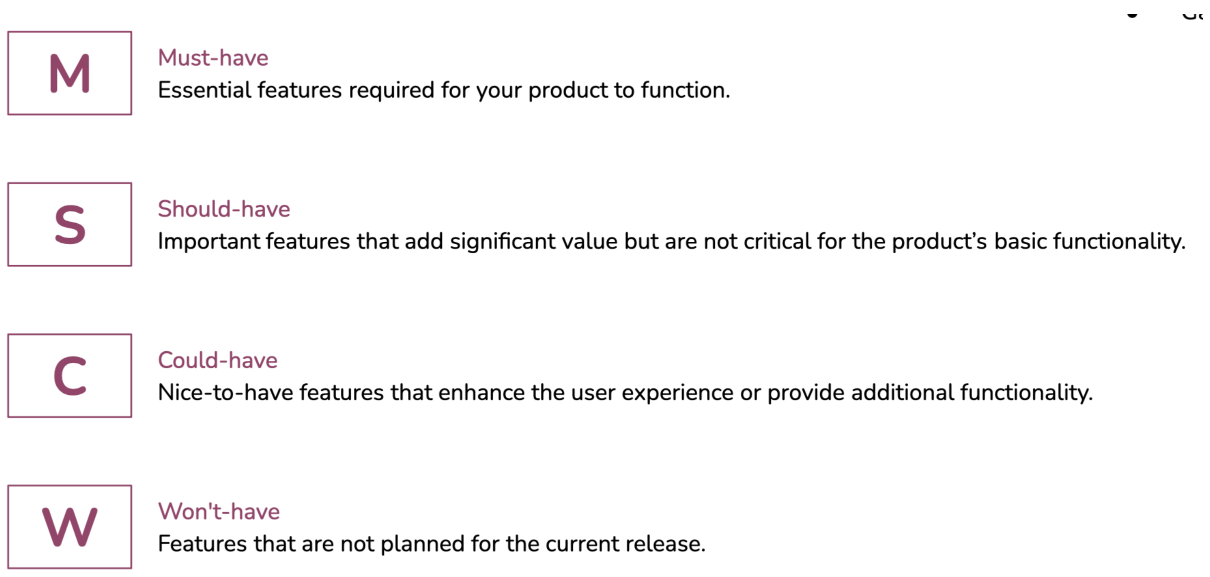
$$\frac{(\text{Reach} \times \text{Impact} \times \text{Confidence})}{\text{Effort}} = \frac{(500 \times 2 \times 0.8)}{6}$$

RICE score: 133

Since we now have the scores for all components, we can calculate the RICE score, which is approximately 133. Now how should we evaluate this figure? Well, the RICE score doesn't have a universal scale or threshold since it's relative to other features being evaluated. When prioritizing, **focus on the features with the highest scores**. These features should usually be addressed first because they provide the most value compared to the effort needed. If two or more features have similar RICE scores, **look at other factors to decide which to prioritize**. For example, consider whether a feature aligns better with your business goals. Next, ensure that essential features are developed first to support others. Finally, timing is also important, especially if a feature addresses an urgent problem or fits a specific opportunity.

MoSCoW Method

The last prioritization method we'll cover is the MoSCoW method. This approach provides a simple and effective way to categorize features based on their importance and necessity. The name **MoSCoW** stands for **M**ust-have, **S**hould-have, **C**ould-have and **W**on't have. Let's explore each of them in much detail.



M	Must-have Essential features required for your product to function.
S	Should-have Important features that add significant value but are not critical for the product's basic functionality.
C	Could-have Nice-to-have features that enhance the user experience or provide additional functionality.
W	Won't-have Features that are not planned for the current release.

- ***Must-have***

So, Must-have features are **essential features required for your product to function**. Without these, the product fails to meet its core purpose. For example, login authentication in a business application is a clear must-have.

- ***Should-have***

Next, Should-have features are **important features that add significant value but are not critical for the product's basic functionality**. These can be developed once the must-haves are complete. For instance, a password reset option is valuable but not essential for the initial release.

- ***Could-have***

Then we have Could-have features. These are **nice-to-have features that enhance the user experience or provide additional functionality, but they aren't essential**. They can be prioritized if there's extra time or resources. Example could be adding a dark mode option that might be appreciated by users. .

- ***Won't-have***

And finally there are Won't-have features. These features are **not planned for the current release because they are not critical or require resources that could be better used elsewhere**. An example might be integration with a niche third-party service that doesn't align with immediate priorities.

Now, let's walk through a practical example of using the MoSCoW method to prioritize features for an email marketing platform. We will categorize features based on their importance to the platform's success. In other words, **we need to focus on what should be developed now versus what can wait for later releases**.

On the slide, you can see the list of features we're considering for this platform.

- *Email Template Builder*
- *A/B Testing Capability*
- *Social Media Integration*
- *Gamification Features (e.g., Achievement Badges for Campaign Milestones)*

Let's start with **Email Template Builder**. That is clearly a **Must-have** because it serves as the foundation of the platform and allows users to create and customize emails. Without this, the platform would not fulfill its primary purpose.

Next, the **A/B Testing Capability** falls under **Should-have**. This feature allows marketers to compare different email versions and optimize performance. However, A/B testing is not critical for the platform's basic functionality, so it can be prioritized right after core features are ready.

Moving on, **Social Media Integration** is a **Could-have** feature. Integrating with social media can extend the platform's capabilities and broaden its appeal, but it doesn't directly impact core email functionality. As a result, this feature can be deferred until the main features are in place.

Finally, we have **Gamification**, like achievement badges. This can make the platform more engaging and rewarding for users, but it doesn't impact core functionality or campaign effectiveness. This feature can be added in the future to boost user engagement.