Product Management

KEY TERMINOLOGIES

BY RUTUJA PATOLE

Product Strategy

A product strategy is a high-level plan describing what a business hopes to accomplish with its product and how it plans to do so. The strategy should answer key questions such as who the product will serve (personas), how it will benefit those personas, and the company's goals for the product throughout its life cycle.

Product management expert Roman Pilcher suggests a strategy should contain the following key elements:

- The market for the product and the specific needs it will address.
- The product's key differentiators or unique selling proposition.
- The company's business goals for the product.

It acts as the guiding star for all product-related decisions, ensuring that each step taken contributes towards the overarching goal of the product. Without it, efforts can become disjointed, leading to wasted resources and a product that fails to resonate with its intended audience.

Product Differentiation

Differentiating a product involves making it stand out from the competition in ways that appeal to a specific market segment. This differentiation focuses on a product's distinctive attributes, advantages, or features relative to others in the marketplace.

Key takeaways:

- The elements of differentiation include product design, marketing, packaging, and pricing.
- A product differentiation strategy should demonstrate that a product has all the features of competing choices but with additional exclusive benefits no one else offers.
- Companies gain a competitive advantage and market share through product differentiation.
- Product differentiation increases market competition and controls prices for consumers.

Go-To-Market Strategy

A go-to-market (GTM) strategy is a step-by-step plan designed to bring a new product to market and drive demand. It helps identify a target audience, outline marketing and sales strategies, and align key stakeholders.

I've seen two major methods for developing a go-to-market strategy: the funnel and the flywheel. While the traditional, one-off funnel method focuses on attracting leads and nurturing them into sales, the flywheel approach uses inbound marketing and other strategies to build long-lasting customer relationships.

Some of the most common benefits of compiling an effective GTM strategy include:

Gaining a comprehensive understanding of the marketplace, the target market, and the proposed product's place in it.

Keeping marketing costs down by identifying promotional channels with the highest return on investment (ROI).

Troubleshooting product positioning and messaging before going to market.

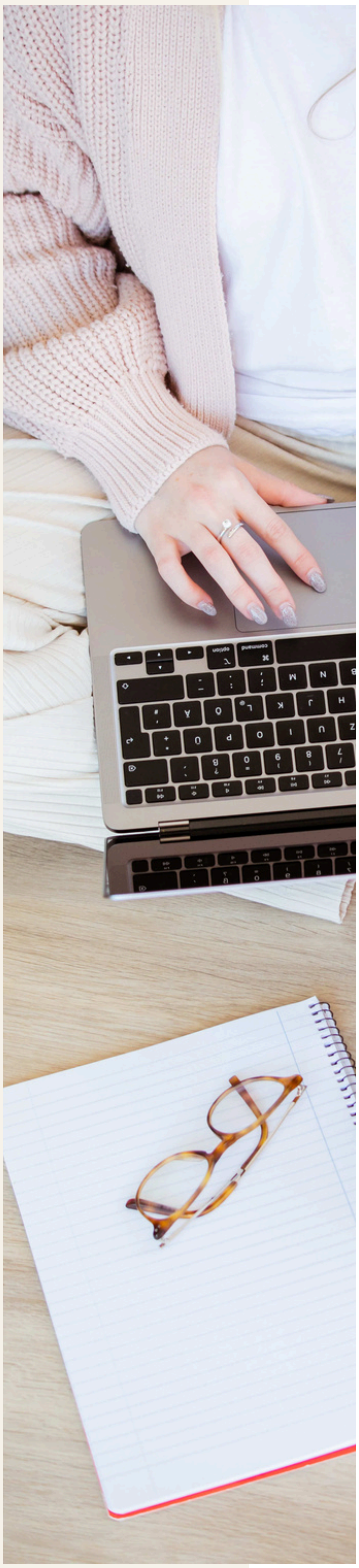
Concretely defining the logistics of distribution and sales channels before launch to ensure maximum market impact.

Value Proposistion

1. This is the value that an entity promises to its consumers if they choose to purchase its products. Value propositions detail the benefits of a product in comparison to the competitor's product, how the company operates, what it stands for and why it deserves the market.

2. Examples of value proposition in renown and successful brands
   - Uber- The smartest way to get around- This capitalizes on efficiency and time saving.
   - Grammarly- Great writing, simplified
   - Stripe- Payments infrastructure for the internet

3. Value proposition communicates the benefits of a product or service in comparison to those offered by the competitors, how the company operates, what it stands for and why it deserves the market. On the other hand, differentiation communicates the unique qualities of the product over the competitor products and highlights the company's product as superior.



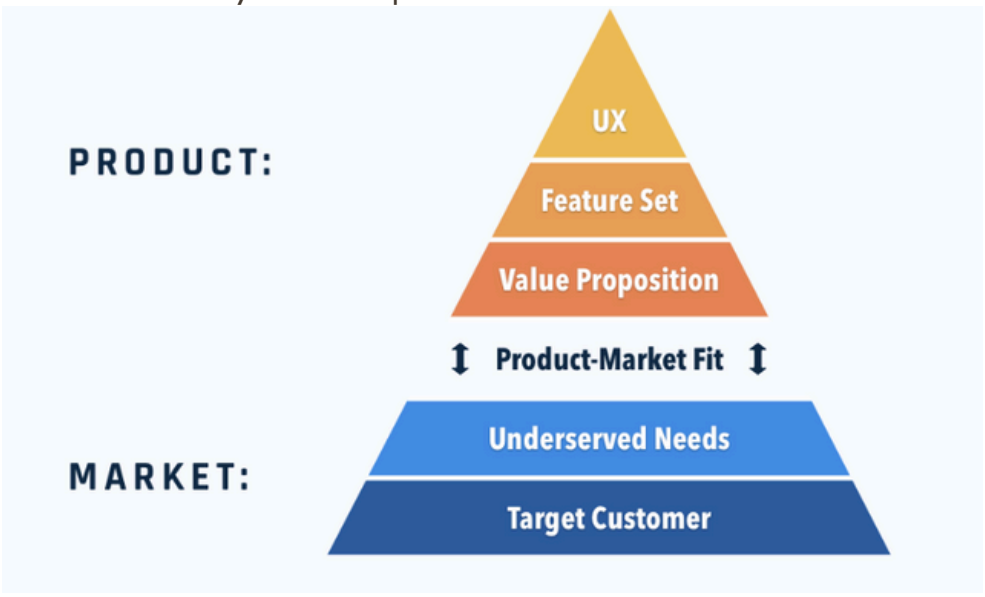
Product–Market Fit

Product–market fit is achieved when a product more effectively fulfills a substantial market need than its competitors, demonstrating its capability to meet genuine consumer demands.

How is Product–Market Fit Measured?

No single set of metrics can tell any business when it achieves product-market fit. But, Andrew Chen, a venture capitalist, offers some signals that a company is heading in the right direction with its offering:

- When surveying potential customers or allowing them to test your product, does some segment indicate they will switch to your product?
- Are some users who have rejected similar products on the market willing to try yours?
- When user testing, do people group your product accurately with the right competitive offerings?
- Do users demonstrate an understanding of your product's differentiators or unique value proposition?
- How do your underlying metrics (such as retention rates of users) measure up against those of your competitors?



Product Roadmap

A product roadmap is a high-level visual summary that maps out the vision and direction of your product offering over time. A product roadmap communicates the why and what behind what you're building. A roadmap is a guiding strategic document as well as a plan for executing the product strategy.

A roadmap for a freshly-minted MVP differs significantly from a mature product on many fronts:

- Horizon: Startups have a much harder time predicting future requirements and opportunities for the products. Therefore their roadmaps probably won't go too far in the future (or if they do it's with some very large asterisks). Established products can make firmer longer-term plans. They have a better understanding of their customers and the market.
- Frequency: When you're young and scrappy, you need to "always be shipping." More mature products can space out their releases with less urgency.
- Dependencies: Startups can move quickly and break stuff. Mature products have a legacy to worry about, third-party integrations to maintain, and regression issues to contend with.
- Goals: A startup is trying to prove its viability, gain traction, & grow. A enterprise product will have nuanced strategic objectives & more targets.

Minimum Viable Product

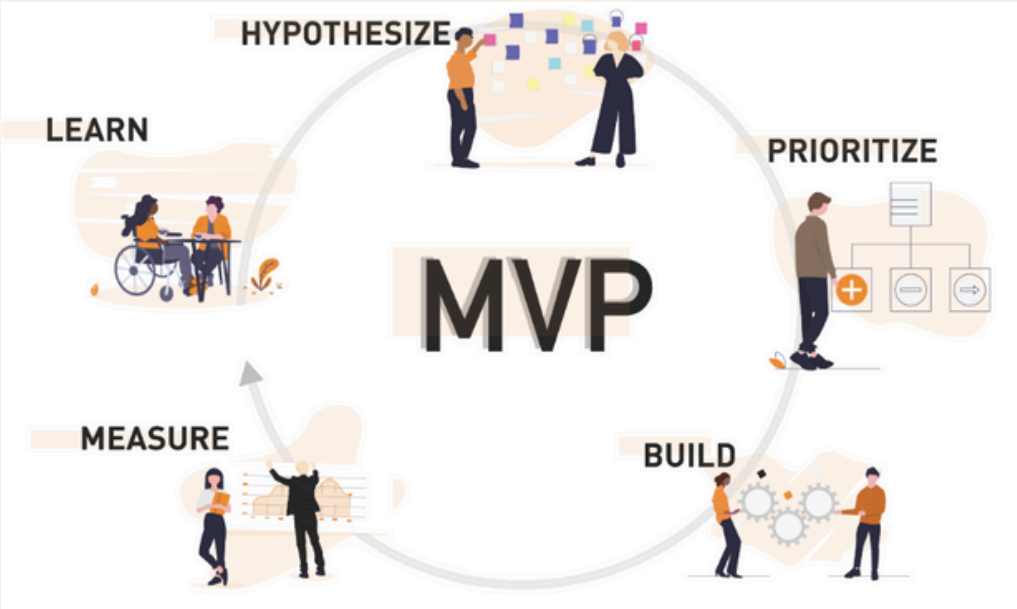
1.

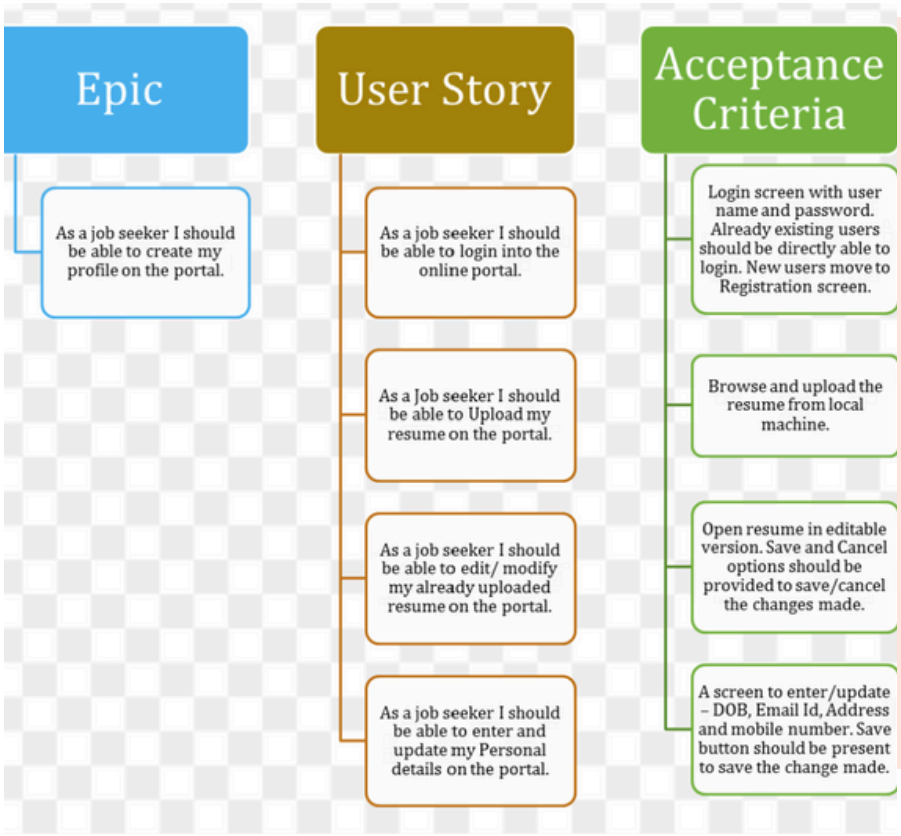
2.

An MVP represents the **simplest form of a product** that can be launched to initial users.

With no money to build a business, the founders of Airbnb used their own apartment to validate their idea to create a market offering short-term, peer-to-peer rental housing online. They created a minimalist website, published photos and other details about their property, and found several paying guests almost immediately.

Why it's important: The concept of an MVP is fundamental in today's fast-paced market environments. This approach enables businesses to adapt and refine their products swiftly, avoiding the pitfalls of dedicating too much time and resources to unproven ideas. This approach speeds up innovation and significantly reduces the risk associated with new product development.





Acceptance Criteria

Client Accepted, Definition of Done (DoD)

Acceptance criteria are a set of conditions that software must meet in order to be accepted by a customer or stakeholder.

Acceptance criteria do not focus on “how” a solution is reached or “how” something is made. Instead, they illuminate the “what” of the work you are doing. For example, the criteria may be:
Users can pay with Google Pay or Apple Pay at checkout.

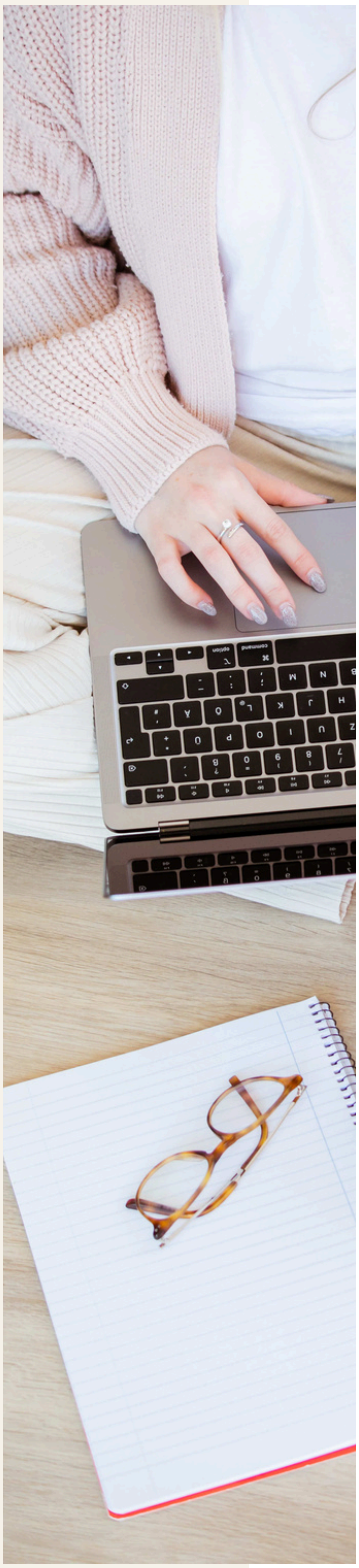
The spirit of acceptance criteria is not to tell you how to do it, for example:
Install a Wordpress plugin that allows you to create a checkout page.
Or –
Write HTML that makes it possible to pay with Apple Pay or Google Pay.
These statements get at how the work will be done, not the conditions for accepting the work. It’s up to the developers on the scrum team to decide the **how** of fulfilling the acceptance criteria.

Alpha Testing

1. Alpha testing is the **first end-to-end testing** of a product to ensure it meets the business requirements and functions correctly. It is typically performed by internal employees and conducted in a lab/stage environment. An alpha test ensures the product really works and does everything it's supposed to do.

2. Alpha tests can also be conducted using both "white box" and "black box" methods. In a white box setting, testers can "look inside" the product to see what's happening during the testing, which is typically not possible in a production setting, while a black box test simply provides the inputs and confirms the outputs are returned as expected.

3. The primary difference between an alpha test and a beta test is who is doing the testing—alpha tests are typically performed by internal employees in a lab or stage environment, while beta tests are conducted by actual users in a production setting. The goal of the alpha test is to catch as many issues as possible before the product has any public exposure or usage.

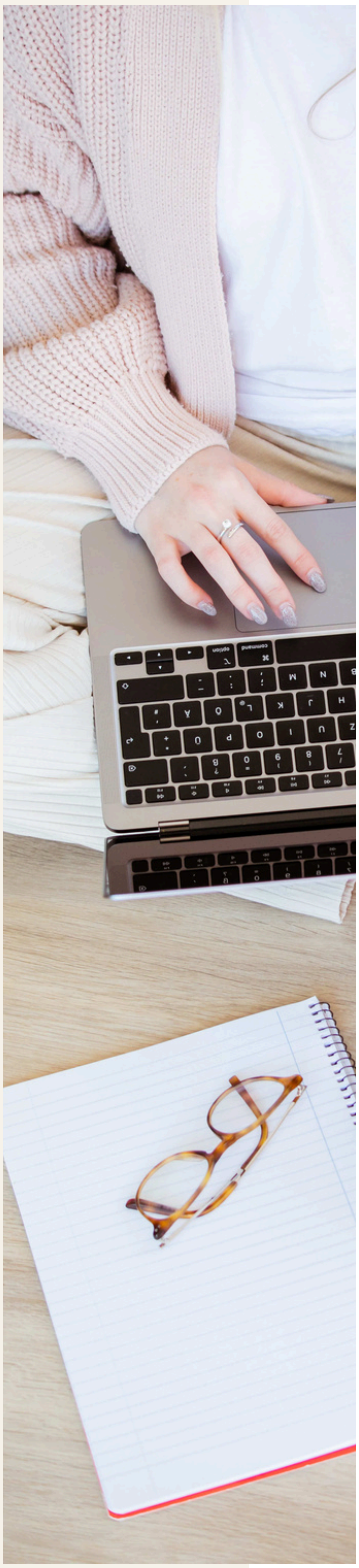


Beta Testing

1. Beta testing is the **final round of testing** before releasing a product to a wide audience. The objective is to uncover as many bugs or usability issues as possible in this controlled setting.

2. Beta testers are “real” users and conduct their testing in a production environment running on the same hardware, networks, etc., as the final release. This also means it’s the first chance for full security and reliability testing because those tests can’t be conducted in a lab or stage environment.

3. Beta tests can either be open or closed. In an open test, anyone can use the product and is usually presented with some messaging that the product is in beta and given a method for submitting feedback. In closed beta, the testing is limited to a specific set of testers, which may be composed of current customers, early adopters, and/or paid beta testers. Sometimes they are conducted by diverting a certain percentage of users to the beta site instead of the current release.



Burndown Chart

Release Burndown Chart, Sprint Burndown Chart

A burndown chart helps agile project management teams keep track of what's been done, what needs to be done and how much time is left in the project.

How to Read a Burndown Chart?

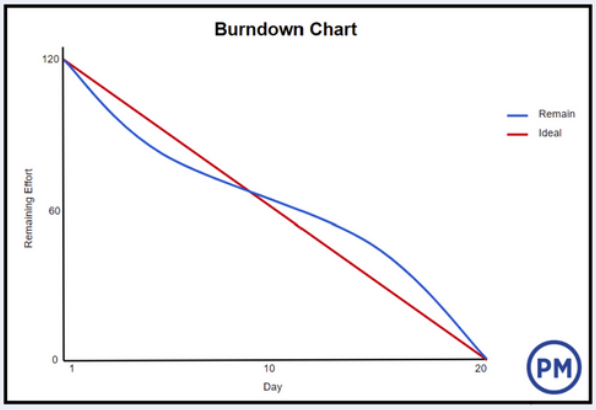
A burndown chart has two axes, x and y. The horizontal axis represents time while the vertical axis displays user story points.

If the actual work line is above the ideal work line, it means there's more work left than originally thought. In other words, the project is behind schedule.

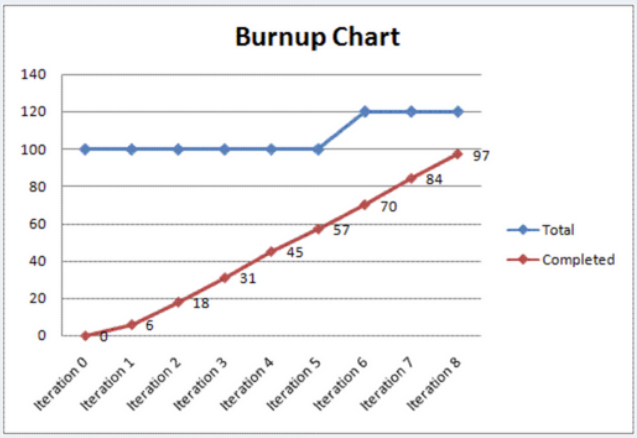
However, if the actual work line is below the ideal work line, there is less work left than originally predicted and the project is ahead of schedule.

A burndown chart and a burnup chart are very similar—they have the same components, achieve the same purpose and are used for agile project management. But there's one major difference.

On one hand, the burndown chart keeps track of the remaining work by removing user stories from the vertical axis as they're completed while the burnup chart adds user stories to the vertical axis as they get done.



Burndown Chart Example



Competitor Analysis

Assessing the strengths and weaknesses of similar products from direct or indirect competitors. Keep this analysis in mind when defining your unique value provisions!

FEATURES	EVENMORE	EVENTBRITE	MEETUP
PERSONALIZED EVENT RECOMMENDATIONS	✓	✓	
FREE EVENT CREATION	✓	✓	✓
DEDICATED MARKETING PLATFORM	✓		
EVENT ANALYTICS & INSIGHTS	✓	✓	✓
CUSTOMIZABLE BUSINESS PROFILES	✓		
USER EVENT EXPERIENCES & REVIEWS	✓		

Competitive analysis framework

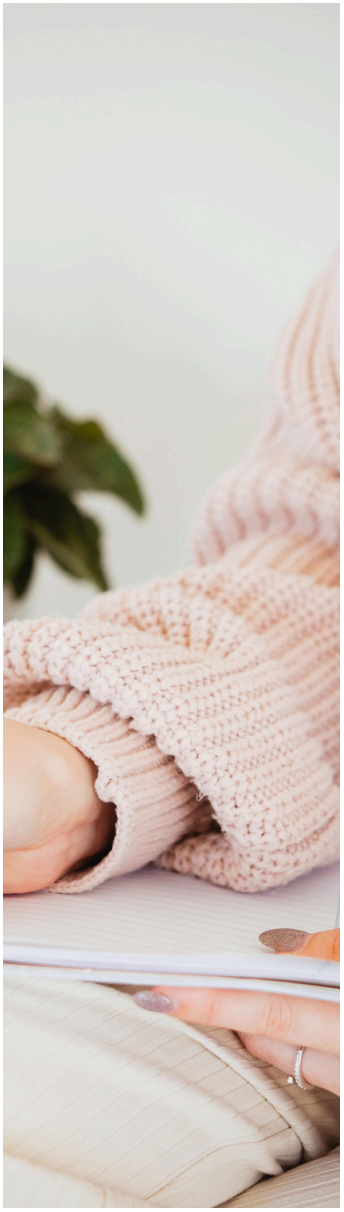
	Your Company	Competitor 1	Competitor 2
Product/service	SEO	SEO/Paid ads	SEO/Website design
Market share	25%	40%	35%
Growth	6%	12%	8%
Target audience	Dentists	Dentists	Dentists
Price structure	Monthly fee	Hourly	Project-based
Marketing strategies	Email/Blog	Email/Blog/ Social media	Social media/ Email/Paid ads
Customer satisfaction	★★★★	★★★★	★★★
Strengths	All-inclusive/ one fee	Brand visibility	Package deals
Weaknesses	Startup with less resources	Expensive	Questionable customer service
Key advantage	Strong values and mission	Industry leader	Highly skilled team

How to do a competitor analysis

- 1. Find out who your competitors are.
- 2. Analyze your competitors and their business structures.
- 3. Evaluate your competitors and their value propositions.
- 4. Evaluate your competitors' marketing efforts.
- 5. Audit your competitors' brand identities.
- 6. Follow each competitor's customer journey.
- 7. Examine audience engagement.
- 8. Conduct a SWOT analysis of your competion.

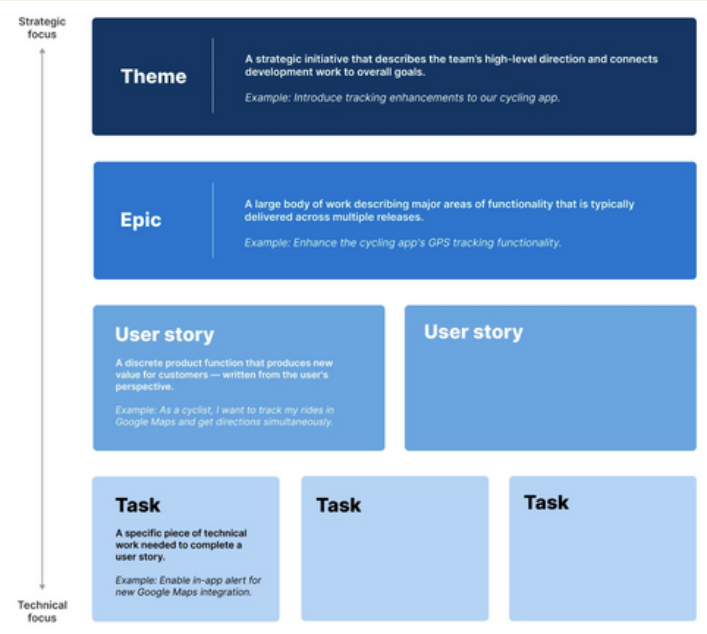
How to use competitor analysis - your next step is to apply the insights to your business

- What product features can you add that differentiate your product from competitors?
- What pricing strategy can you use to attract new customers to your offerings?
- What design features can you add to your brand to make it stand out?
- How can you compose a value proposition that stands out from competitors? We help [target customer] do [outcome, benefit, experience] by doing / offering [product or service].
- How can you design a more seamless, frictionless customer journey that leads consumers to make a purchase and become loyal brand ambassadors?
- What approaches can you take on marketing channels where your competitors have a presence to distinguish your messaging and present your offerings as the best choice?

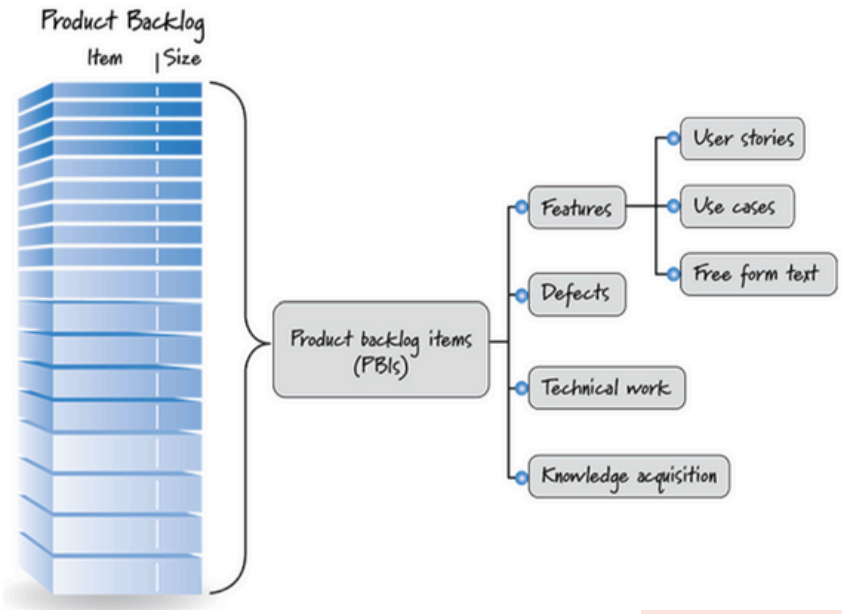


Theme, Epic & User Stories

- The **roadmap** communicates the product plan and aligns the whole organization around it. The roadmap shows how the product will evolve over time, and usually includes a timeline, feature releases, and goals.
- **Themes** are long-term strategic objectives with a broader scope. They provide context for decision-making and help navigate the product strategy within the organization. Agile themes sit on top of the work breakdown hierarchy and drive the creation of epics.
- **Epics** are collections of tasks or user stories. Epics break down development work into shippable components while keeping the daily work connected to the larger theme. Epics are more specific than themes and can be measured so that PMs can observe their contribution to the organization's overall goal.
- **User stories** are the smallest piece of work in the agile framework. A user story is a brief explanation of a product feature written from the end user's perspective that articulates how the user will experience value. Some organizations may classify larger user stories (stories that can't be delivered within a single sprint) as epics. Alternatively, larger stories could be broken down into sub-tasks.



Product Backlog



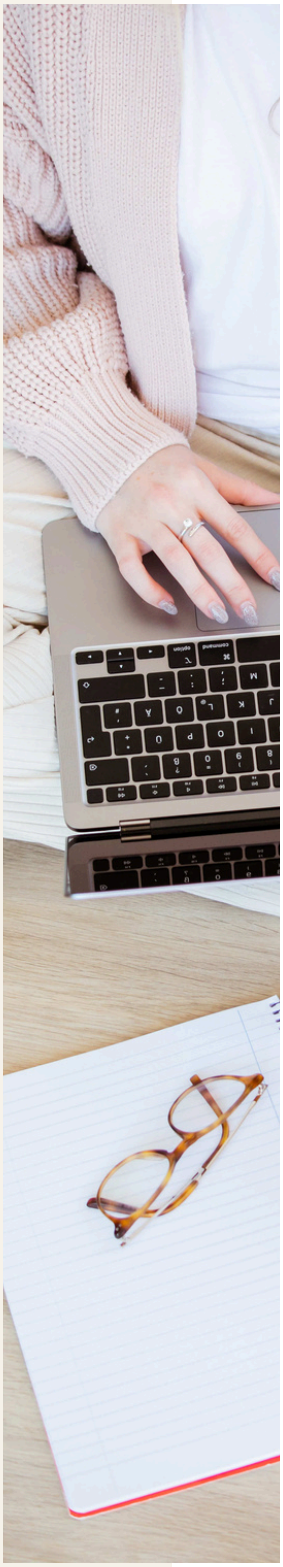
A product backlog is a prioritized list of work for the development team that is derived from the product roadmap and its requirements. The most important items are shown at the top of the product backlog so the team knows what to deliver first. Grooming is made up of three principal activities: creating and refining PBIs, estimating PBIs(size), and prioritizing PBIs.

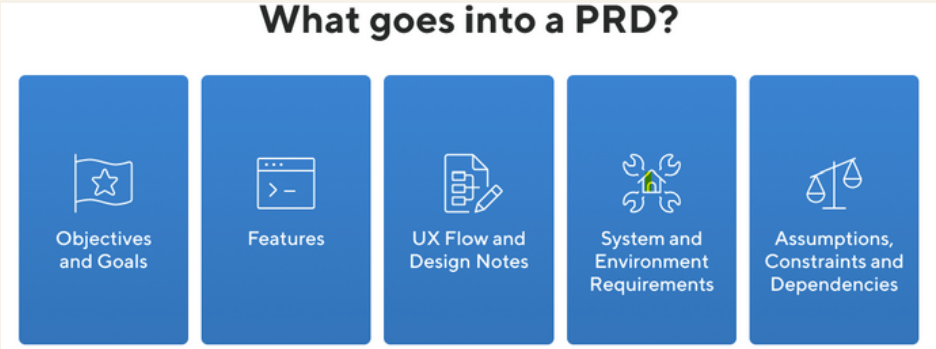
- The **product strategy** is a high-level overview of how the company vision will be achieved.
- The **product roadmap** dictates how the strategy will be executed.
- The **product backlog** contains the task-level details required to develop the product as outlined in the roadmap.

Product backlog	Sprint backlog
Product owner has control over the product backlog	Development team owns each sprint backlog
Focused on the entire product vision	Focused on fulfilling the sprint goal
No time limit to complete the entire backlog	Completed in a designated time period
Independent entity that changes until product completion	A subset of the product backlog with defined tasks

Product Requirements Documents

1. A product requirements document (PRD) is an artifact used in the product development process to communicate what capabilities must be included in a product release to the development and testing teams. Refer for an example of PRD: https://www.atlassian.com/agile/product-management/requirements



2. 

3. The PRD may follow on the heels of a marketing requirements document (MRD)—created by product marketing, marketing, or product management as well—that describes customer demand, market opportunity, and a business case for the overall product or a particular product release.Based on the PRD, a number of other artifacts will be created by others in the organization. Engineering will create a functional specification, which describes how each item in the PRD will be implemented, and they may also create (or update) an architectural design document. UX will create wireframes and mockups as needed, and quality assurance will write a test plan ensuring every single use case in the PRD can be successfully executed during testing.

Software Development Life Cycle

The software development lifecycle (SDLC) is the cost–effective and time–efficient process that development teams use to design and build high–quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond.

How does SDLC work?

The development process goes through several stages as developers add new features and fix bugs in the software.

The **planning phase** typically includes tasks like cost-benefit analysis, scheduling, resource estimation, and allocation. The development team collects requirements from several stakeholders such as customers, internal and external experts, and managers to create a software requirement specification document.

In the **design phase**, software engineers analyze requirements and identify the best solutions to create the software. For example, they may consider integrating pre-existing modules, make technology choices, and identify development tools. They will look at how to best integrate the new software into any existing IT infrastructure the organization may have.

In the **implementation phase**, the development team codes the product. They analyze the requirements to identify smaller coding tasks they can do daily to achieve the final result.

The development team combines **automation and manual testing** to check the software for bugs. Quality analysis includes testing the software for errors and checking if it meets customer requirements. Because many teams immediately test the code they write, the testing phase often runs parallel to the development phase.

Deploy Phase When teams develop software, they code and test on a different copy of the software than the one that the users have access to. The software that customers use is called production, while other copies are said to be in the build environment, or testing environment.

In the **maintenance** phase, among other tasks, the team fixes bugs, resolves customer issues, and manages software changes. In addition, the team monitors overall system performance, security, and user experience to identify new ways to improve the existing software.

What are SDLC models?

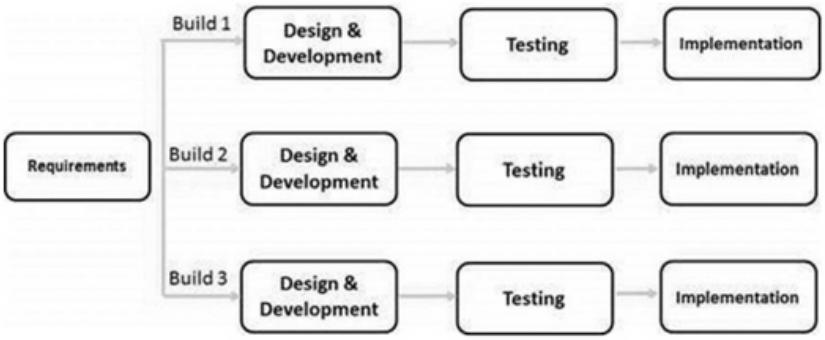
A software development lifecycle (SDLC) model conceptually presents SDLC in an organized fashion to help organizations implement it. Different models arrange the SDLC phases in varying chronological order to optimize the development cycle. We look at some popular SDLC models below.

The **waterfall model** arranges all the phases sequentially so that each new phase depends on the outcome of the previous phase. Conceptually, the design flows from one phase down to the next, like that of a waterfall.
Projects based on the waterfall model are well defined, predictable and have specific documentation. They also have the following characteristics:

- fixed requirements;
- ample resources;
- an established timeline;
- well-understood technology; and
- unlikely to require significant changes.
- The waterfall model provides discipline to project management and gives a tangible output at the end of each phase. However, there is little room for change once a phase is considered complete, as changes can affect the software's delivery time, cost, and quality. Therefore, the model is most suitable for small software development projects, where tasks are easy to arrange and manage and requirements can be pre-defined accurately.
- Safety-Critical Systems: The Waterfall Model is often used in the development of safety-critical systems, such as aerospace or medical systems, where the consequences of errors or defects can be severe.
- Government and Defense Projects: The Waterfall Model is also commonly used in government and defense projects, where a rigorous and structured approach is necessary to ensure that the project meets all requirements and is delivered on time.
- Projects with well-defined Requirements: The Waterfall Model is best suited for projects with well-defined requirements, as the sequential nature of the model requires a clear understanding of the project objectives and scope.

SDLC–Iterative Model

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).



Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development. "During software development, more than one iteration of the software development cycle may be in progress at the same time." This process may be described as an "evolutionary acquisition" or "incremental build" approach."

In this incremental model, the whole requirement is divided into various builds. During each iteration, the development module goes through the requirements, design, implementation and testing phases. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement.

- **Adv**- Parallel development can be planned. Testing and debugging during smaller iteration is easy. Better suited for large and mission-critical projects. It supports changing requirements.
- **Disadv**- More resources may be required. Not suitable for smaller projects. End of project may not be known which is a risk.
- **Applications**- A new technology is being used and is being learnt by the development team while working on the project. Resources with needed skill sets are not available.