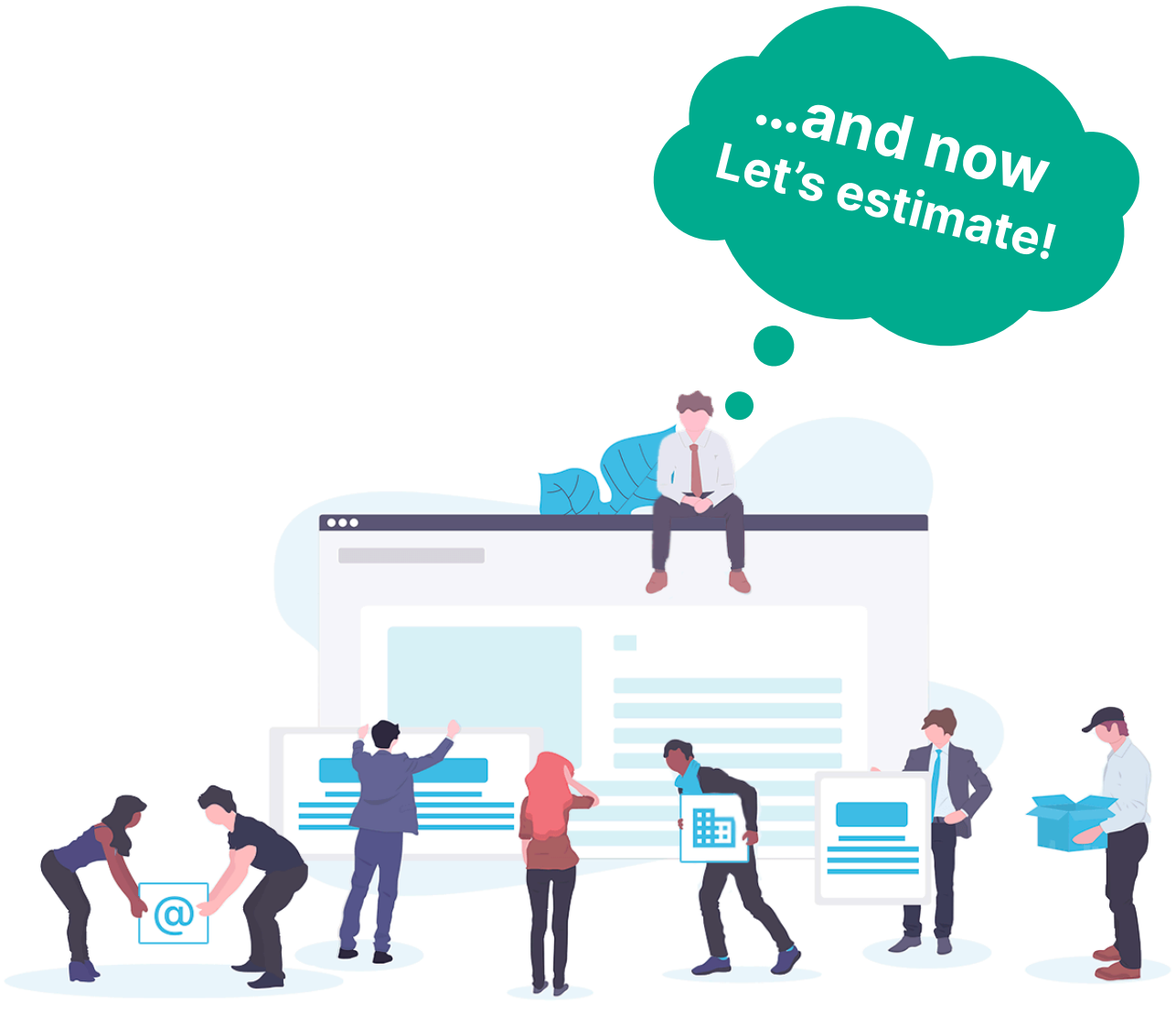
Stories On Board

Backlog Estimation Handbook

Comprehensive Guide to Estimation Techniques



Backlog estimation made simple

Backlog estimation is the process of evaluating the time, effort, or complexity required to complete tasks (user stories) in a product backlog. These tasks represent future work to be done in a project, and estimating them is crucial for sprint planning, prioritization, and resource allocation. Common estimation techniques include Planning Poker, Story Points, and T-Shirt Sizing, each offering a way to assess tasks based on effort, complexity, or time required.

Backlog estimation is important because it helps product teams make informed decisions about what to prioritize, ensuring that higher-value items are addressed first. Accurate estimation also helps with capacity planning, enabling teams to set realistic sprint goals and deadlines, which improves efficiency and predictability in product delivery. Ultimately, backlog estimation supports better communication, alignment, and transparency between product owners, development teams, and stakeholders.

What are the benefits of backlog estimation?

Estimation enables agile teams to balance flexibility with predictability, supporting a continuous flow of work that aligns with business goals and user needs.

Improved Planning

Estimation helps teams plan sprints and releases with more accuracy. By understanding the effort and complexity of tasks, teams can make informed decisions about what to prioritize and how much can realistically be accomplished within a sprint.

Better Prioritization

Estimations give clarity on what tasks are critical and require immediate attention versus those that can be scheduled for later. This helps teams focus on delivering value early and often.

Enhanced Communication

Estimating backlog items fosters better communication between product owners, scrum masters, and development teams. It encourages discussions about scope, expectations, and potential risks, leading to more alignment and fewer misunderstandings.

Managing Team Capacity

Knowing the estimated size of backlog items allows teams to better manage their workload and avoid over-committing. Teams can track progress more effectively, ensuring steady delivery and avoiding burnout.

What are the most common estimation techniques?

➜ Planning Poker (Scrum Poker)

Planning Poker is a collaborative estimation technique where team members independently select estimates for a backlog item using a set of cards with pre-defined values (often based on Fibonacci sequence: 1, 2, 3, 5, 8, 13, 21, etc.). Afterward, the team discusses their selections and comes to a consensus.

How it works

- Each team member selects a card representing their estimate of the effort required.
- If estimates vary significantly, the team discusses the reasoning behind the estimates.
- The process repeats until consensus is reached.

Example

Backlog item: *"Create user login functionality."*

- Developer A estimates 5 story points (based on past experience with similar features).
- Developer B estimates 8 story points (thinking the security integration might be complex).
- After discussion, the team agrees on 5 story points by concluding the security aspect can be handled more easily than initially thought.

T-Shirt Sizing

This method uses clothing sizes (XS, S, M, L, XL) to estimate the relative size or effort of tasks. It's often used for rough or early estimations before a detailed breakdown of the work is available.

How it works

- Each backlog item is assigned a size, with larger sizes indicating more complex or time-consuming tasks.
- It focuses on relative effort rather than precise time.

Example

Backlog item: *"Add search functionality to the app."*

- Team discusses and assigns an "L" (large) size to the search feature based on its complexity (filters, multiple categories, etc.).
- For a simpler task like *"Create a contact form,"* the team assigns an "S" (small) size.

Story Points

Story points measure the effort or complexity of a backlog item, usually on a relative scale. The Fibonacci sequence is often used here to represent increasing complexity in non-linear increments. Story points focus on effort, not time, considering factors like risk, complexity, and uncertainty.

How it works

- Team members evaluate a user story's effort compared to other stories.
- Story points often follow the sequence: 1, 2, 3, 5, 8, 13, etc.

Example

Backlog item: *"Implement email notifications for user actions."*

- The team discusses the effort required and assigns 3 story points to this item, as it's fairly straightforward.
- Meanwhile, a more complex task like *"Implement real-time chat functionality"* might get assigned 8 story points due to its complexity and integration challenges.

Fibonacci Sequence

The Fibonacci sequence (1, 2, 3, 5, 8, 13, etc.) is commonly used for assigning story points. The idea behind using these numbers is that tasks become more difficult to estimate accurately as they grow in complexity, so gaps between numbers widen.

How it works

- Team members assign story points from the Fibonacci series to each backlog item.
- This helps differentiate between small, medium, and large tasks and accounts for increased uncertainty with larger tasks.

Example

Backlog item: *"Develop a feature for exporting data to CSV."*

- The team evaluates this task and assigns it 5 story points (medium complexity).
- For another task, "*Develop a custom report generator,*" the team assigns 13 points due to the increased complexity and potential unknowns.

Affinity Estimation

Affinity estimation involves sorting backlog items into categories based on their relative sizes or effort. It's a fast, low-detail method useful for quickly organizing large sets of stories.

How it works

- Teams categorize stories by grouping similar-sized items together.
- Team members place user stories on a scale (e.g., small to large or easy to complex).

Example

Backlog items include :

"Add social media sharing,"
"Create a dashboard,"
"Integrate payment system."

- The team places *"Add social media sharing"* in the small effort category, *"Create a dashboard"* in medium, and *"Integrate payment system"* in large due to the complexity of third-party API integrations.

Bucket System

The Bucket System divides backlog items into "buckets" or pre-defined bins with increasing estimates. Team members place items into these buckets based on their perceived complexity.

How it works

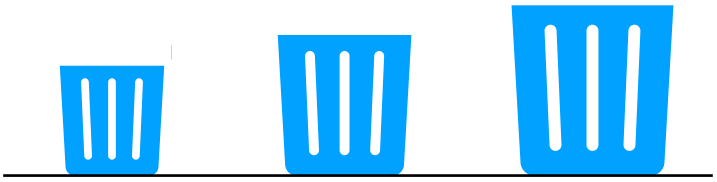
- Buckets represent different complexity levels (e.g., 1, 2, 3, 5, 8, etc.).
- Team members collaboratively assign tasks to buckets.

Example

Backlog items are listed as:

> *“Implement password reset,"*
> *“Design new homepage,"*
> *"Set up user roles."*

- The team assigns *"Implement password reset"* to bucket 3, *"Design new homepage"* to bucket 8 (larger scope), and *"Set up user roles"* to bucket 5.



Dot Voting

This technique is more about prioritization but can indirectly help with estimation. Team members are given dots (votes) to allocate across different backlog items, signaling which items they believe should be prioritized or require significant effort.

How it works

- Each team member is given a set number of dots to "vote" on the tasks they believe are most important or complex.
- Items with the most votes get attention first.

Example

A team is considering three backlog items:

> *"Enable profile picture uploads,"*
> *"Set up password complexity requirements,"*
> *"Implement user notification system."*

- Most team members place dots on *"Implement user notification system,"* indicating that this is likely the most complex or highest-priority task.

Three-Point Estimation

This method involves estimating effort using three data points: the most optimistic estimate (O), the most pessimistic (P), and the most likely (M). The final estimate is calculated using a weighted average of these points.

How it works

- Formula: Estimate = (O + 4M + P) / 6
- This method accounts for uncertainty and variability in estimates.

Example

Backlog item: *"Implement mobile-responsive layout."*

The team estimates:

- **Optimistic:** 5 hours
- **Most likely:** 8 hours
- **Pessimistic:** 12 hours

Using the formula, the estimate = (5 + 4(8) + 12) / 6 = 8.16 hours.

Team Estimation Game

In the Team Estimation Game, team members estimate the complexity of each backlog item by arranging them on a scale, discussing and re-ordering them until consensus is reached. This method is great for large backlogs and encourages open discussion.

How it works

- Teams place items on a scale from easiest to hardest.
- The team re-evaluates the placement until all agree on the order.

Example

Tasks like "*Update user profile page*" and "*Implement chat feature*" are sorted relative to each other.

- "*Update user profile page*" is placed lower on the complexity scale, while "*Implement chat feature*" moves toward the higher end.

Relative Sizing (Comparative Estimation)

In this method, backlog items are estimated by comparing them to previously completed tasks. It helps teams estimate new stories based on experience.

How it works

- Teams look at previously completed tasks and compare them with new tasks.
- Each new task is given a size relative to the completed ones.

Example

A previous task, "*Create a login form*," took 3 story points.

- The team compares the new task, "*Create a registration form*," and assigns it 5 story points due to added complexity like email verification.

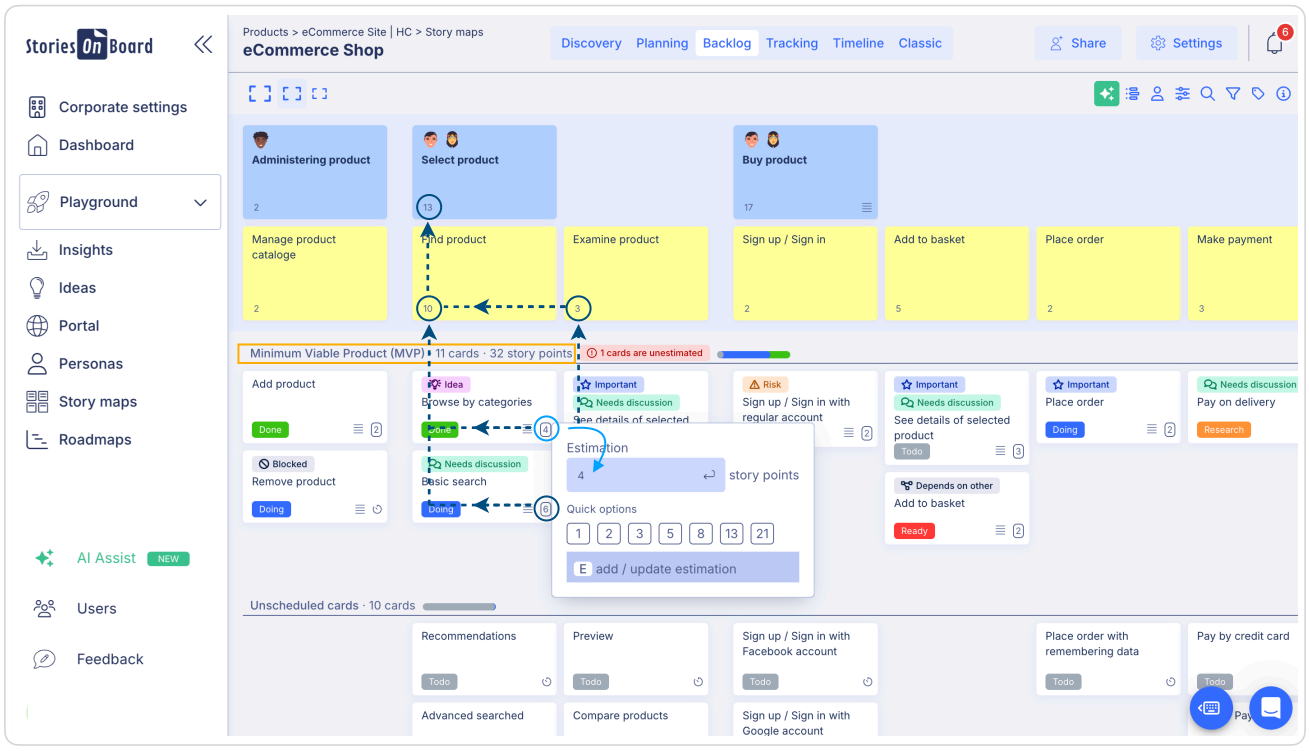
Story points estimation on a story map

Story points estimation on a story map is a useful approach that integrates the visual structure of user story mapping with the agile practice of relative estimation. By estimating user stories on the map using story points, teams can gain insights into the overall complexity and effort required to complete different parts of a product.

How to use StoriesOnBoard for user story points estimation

Visual Context

On the story map, user stories are arranged along the user's journey, allowing teams to see not only individual tasks but also how those tasks fit into the bigger picture. Adding story points to these stories within the context of the map helps teams balance complexity across the full product flow, rather than in isolation.



Relative Estimation

Since story points are used to represent the relative effort of a task (not absolute time), you can compare user stories directly within the map. For instance, if one story on the map has been estimated at 3 points and another at 8, the team can prioritize based on both value and effort.

Prioritization and Sprint Planning

With story points assigned to each story on the map, teams can more easily identify the highest-value, lowest-effort tasks to include in the next sprint, accelerating MVP development or incremental feature releases.

Capacity Planning

By summing the total story points within a section of the story map, teams can estimate how many sprints or how much capacity is needed to complete a specific goal, aligning planning with team velocity.

Story points estimation on a story map is a useful approach that integrates the visual structure of user story mapping with the agile practice of relative estimation. By estimating user stories on the map using story points, teams can gain insights into the overall complexity and effort required to complete different parts of a product.

Learn more about StoriesOnBoard

StoriesOnBoard is lightweight end-to-end product management platform helping product teams build amazing products that users love, including:

- AI powered user story mapping
- Roadmaps and prioritization frameworks
- Feedback management (insights, ideas, portals)

www.storiesonboard.com

Conclusion

Choosing the right estimation technique depends on your team's experience, the complexity of tasks, and project requirements. Techniques like Planning Poker or Story Points are great for agile teams that need relative estimation, while methods like Three-Point Estimation can add precision where needed. The key is to engage the team in a discussion, ensuring that all members have a shared understanding of the effort required, leading to better sprint planning and backlog management.