# WRITING USER STORIES

Summarizing what customers or users want to be able to do; used to bridge design research with defining requirements for software development.

| | |
|---|---|
| **Duration** | 0.5–5 days (depending on complexity and amount of data) |
| **Physical requirements** | Research wall or any other form of accessible research data, personas, journey maps, system maps, paper, pens, masking tape |
| **Energy level** | Low |
| **Researchers/Facilitators** | Minimum 1 (a better approach is to have teams of 2–3 researchers) |
| **Participants** | 2–12 with good knowledge of the research data (optional) |
| **Expected output** | User stories |

User stories are used in software development to define requirements from a user or customer perspective, in contrast to often rather product-based requirement documents.[01] User stories can be used in various stages of a design process:

— During research to request non-complex features that could be implemented in a short time without prior prototyping ("quick wins" or "low-hanging fruit"), or to report critical bugs that hinder users from utilizing or signing up for the software.

— During ideation and idea selection to speak the same language as the IT team during co-creative workshops and to break down ideas into actionable features.

01 User stories are used in many agile software development frameworks, such as Extreme Programming, Scrum, and Kanban. Mind that different approaches often use specific templates for how to phrase user stories. See, for example, Schwaber, K., & Beedle, M. (2002). *Agile Software Development with Scrum (Vol. 1)*. Prentice Hall.

OFTEN, **USER STORIES** ARE FORMULATED LIKE THIS:

**As a**

.................................................. *(type of user/persona/role)*,

**I want**

.................................................. *(action)*,

**so that**

.................................................. *(outcome)*.

— During prototyping to quickly agree on which stories need to be part of the first prototype or the MVP, to be able to test selected stories, and to agree in which sequence the following stories should be implemented.

— During implementation to allow seamless integration with an agile development process that is based on user stories, and to be able to quickly adapt and iterate when technical difficulties occur during implementation.

The software requirements can be broken down into a set of user stories.

As an easy example, a user story related to location-based services on your smartphone could be formulated like this: "As a regular customer, I want to get notifications from restaurants I prefer that are nearby, so that I don't have to search."

User stories should be formulated without IT-specific language. Write these as seen from the user's perspective, using simple, concise words, so that everyone can understand them. In service design, user stories are used to connect design research with actionable input for IT development. Often, when a research team identifies potential "quick wins" for existing software, formulating these insights as user stories is all that is needed for an IT team to develop a "hotfix."[02] At a later stage, these user stories can also be used during prototyping and particularly during implementation to turn low-fidelity prototypes into working software.

Just as journey maps have different zoom levels, software requirements also have different scales. A set of user stories can be combined into what is called an "epic" – a longer, rather sketchy story without a lot of details. Epics describe the big picture of what a piece of software can do. Epics are then typically broken down into several user stories over time based on prototyping, user feedback, and research data.

Reformulating the same example regarding a requirement for a location-based service on your smartphone as a job story could look something like this: "When I stroll through a new city around lunch time, I want to be notified when I'm near a restaurant that matches my preferences so I can go there directly instead of searching for it."

This example illustrates the main difference between a user story and a job story. The job story focuses more on the context of a specific use case and does not include a role or persona like a user story. It makes sense to clarify with your IT team if they use a specific framework for user stories, job stories, epics, and so on.

02 A hotfix is a fast solution for an urgent problem in a software product. Usually, a hotfix is deployed to fix a critical software bug.

## Step-by-step guide

**1 Prepare and print out data**
User stories can be created at any moment in a service design process. They are also useful to find gaps in your research data and to formulate further research questions, hypotheses, or assumptions. Use your research wall or prepare your research data by printing out key pictures, writing out great quotes, visualizing audio recordings or videos as quotes or screenshots, and putting out your collected artifacts. Prepare the room with materials, such as paper, sticky notes, pens, and of course your research data, as well as existing personas, journey maps, or system maps. Also, think about who you should invite to write user stories, particularly from your IT department.

**2 Write initial user stories**
Go through your research data and write down initial user stories based on your research findings or patterns you find within your data. If you work in teams, split up into subgroups of 2–3 participants to write user stories. Check your data if you see divergences between what customers expected and what they really had to do. Write down user stories for both scenarios: how a piece of software is working today (mostly product-centered) and how users expected it should work (mostly user-centered). Comparing these two will give you insights on how to improve the software and potentially give you ideas for some quick wins.

**3 Cluster user stories into epics**
Hang up your user stories on a wall and cluster similar ones next to each other. Check if clusters of user stories can be combined into epics. Alternatively, some user stories might be so big that they are epics and should be broken down into smaller user stories. You can merge similar user stories or rephrase them to make clear that they are different. Then try to prioritize them, for example, from a customer's perspective: which of these could have the biggest impact on the customer?

**4 Link user stories to data**
User stories should always be based on research data. Link them to your research data (e.g., by using an indexing system). When you present your user stories, it helps if you add some of your research data to back them up. If possible, prefer first-level constructs as evidences for your key insights, such as photos, videos, or quotes from real people.

AS AN ALTERNATIVE TO USER STORIES, YOU CAN ALSO FORMULATE **JOB STORIES LEVERAGING THE JTBD FRAMEWORK**, SUCH AS:

**When** ................................................................ *(situation/context)*,

**I want to** ................................................................ *(motivation)*,

**so I can** ................................................................ *(expected outcome)*.

**5 Find gaps and iterate**

Are you missing some data for some of your epics and/or user stories? Use these gaps as research questions and iterate your research to fill the gaps with data. Also, consider inviting real customers or employees to review your insights and to give feedback on them.

**6 Follow-up**

Document your progress with photos and write a summary of your user stories in a format that both your team and the IT team can work with. Add evidence from your research data to your epics and user stories. Use an indexing system to link your insights to all the underlying data.

## Method notes

→ Often, teams use a mixed format for user stories that fits their culture and process. If you agree with developers in advance on how you formulate them, and, if possible, even include one or two of their team members in your research team, you'll have a much smoother transition.

→ Although this chapter focuses on software development as the main field of application of user stories, they can also be used beyond software development to define the requirements of any physical/digital product or service. ◂