

User story and Acceptance criteria

What is User Story and Acceptance Criteria?

In the software development industry, the word “**requirement**” defines our goal, what the customers need exactly, and what will make our company increase its business.

The term ‘requirement’ has different names in different project methodologies: In Waterfall, it is referred to as a “Requirement/Specification Document”, in Agile or SCRUM it is referred to as an “Epic” or “User Story”.

Under the Waterfall model, the Requirement documents are huge docs of 200 or more pages as the whole product is implemented in one phase. But this is not the case with Agile/SCRUM because in these methodologies the requirements are given for small functionalities or features as the product is prepared in a step-by-step manner.

What is a User Story?

A user story is a requirement for any functionality or feature that is written down in one or two lines and max up to 5 lines. A user story is usually the simplest possible requirement and is about one and only one functionality (or one feature).

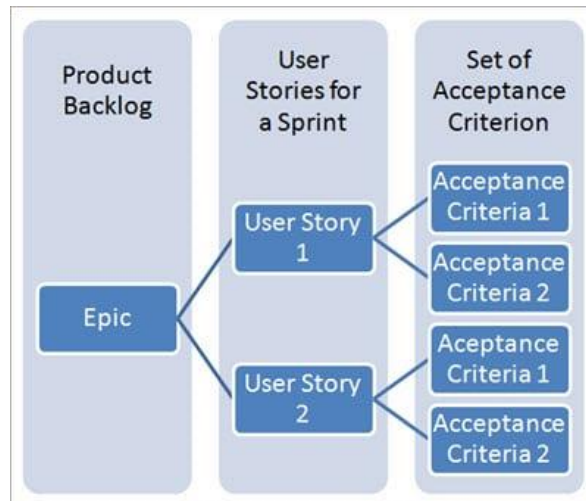
User stories articulate the “who,” “what,” and “why” of a requirement, helping the team understand the desired outcome from the user’s perspective.

The most commonly used standard format for a User Story creation is stated below:

As a <user role/customer>, I want to <goal to be accomplished> so that I can <reason of the goal>.

Example:

As a WhatsApp user, I want a camera icon in the chat write box to capture and send pictures so that I can click and share my pictures simultaneously with all my friends.



Jira Software interface showing the **Backlog** for the **Nucleus** project.

Left Sidebar:

- Nucleus Software project
- Roadmap
- Backlog
- Board
- Code
- On-call
- Project pages
- Add item
- Project settings

Top Navigation:

- Your work
- Projects
- Filters
- Dashboards
- People
- Apps
- Create

Backlog View:

- Search bar
- Filters: Epic, Label, Version
- Epics list: Forms, Feedback, Accounts, Billing, AWS spike
- Sprint 3: 5 issues (3/Jan/19 02:59 PM - 7/Feb/19 02:59 PM)
- Issues list with labels (BILLING, ACCOUNTS, FEEDBACK) and assignees
- + Create issue

Issue ID	Issue Description	Label	Assignee
NUC-335	Affiliate links integration - frontend	BILLING	[Assignee]
NUC-342	Fast trip search	ACCOUNTS	[Assignee]
NUC-341	Quick payment	FEEDBACK	[Assignee]
NUC-340	Account settings defaults	ACCOUNTS	[Assignee]
NUC-339	Billing system integration - frontend		[Assignee]
NUC-337	Multi-dest search UI mobileweb	ACCOUNTS	[Assignee]
NUC-360	Onboard workout options (OWO)	ACCOUNTS	[Assignee]
NUC-344	Optimize experience for mobile web	BILLING	[Assignee]
NUC-338	Multi-dest search UI web	ACCOUNTS	[Assignee]
NUC-354	Shopping cart purchasing error - quick fix required.		[Assignee]
NUC-343	Fluid booking on tablets	FEEDBACK	[Assignee]
NUC-346	Adapt web app no new payments provider		[Assignee]
NUC-336	Quick booking for accommodations - web		[Assignee]

What are the Acceptance Criteria?

The acceptance criteria is a set of accepted conditions or business rules for which the functionality or feature should satisfy and meet, to be accepted by the Product Owner/Stakeholders.

This is a very important part of user story completion, and it should be studied by the Product Owner and Business Analyst very meticulously because missing a single criterion can cost a lot. This is a simple numbered or bullet list.

The format is as follows: *“Given some precondition when I do some action then I expect the result”*.

Example (w.r.t for the above user story):

- Please consider chatting with a friend and I should be able to capture a picture.
- When I click on a picture, I should be able to add a caption to the image before sending it.
- If there is some problem with starting my phone camera, an error message like “Camera could not be started.” etc., should be shown accordingly.

Hence, the User story defines the requirement for any functionality or feature while the Acceptance Criteria defines the “Definition of done” for the user story or the requirement.

Why Are Acceptance Criteria Important?

It acts as a bridge between the user story and the development process. It is important because

- **Clarity and Alignment:** They provide clear and concise guidelines on what needs to be done, ensuring that everyone on the team has a shared understanding of the story.
- **Scope Management:** By defining what is included and what is not, these help manage the scope of a user story, preventing scope creep.
- **Testing and Validation:** They offer a basis for writing test cases and help in validating that the functionality works as expected. This helps in delivering a high-quality product that fulfills the user’s needs.
- **Measurable Success:** Clear acceptance criteria allow for objective evaluation of a user story’s completion. This eliminates subjective interpretations and ensures the delivered functionality meets the intended requirements.
- **User Satisfaction:** They ensure that the final product meets the user’s needs and expectations, leading to higher satisfaction.

Rule-Oriented vs. Scenario-Oriented Acceptance Criteria

Rule-oriented acceptance criteria focus on specific rules or conditions that must always be met for the user story to be considered complete. *Scenario-oriented acceptance criteria* describe specific situations or use cases that demonstrate how the system should behave under certain conditions.

Rule-oriented acceptance criteria are ideal for stories focused on validation, compliance, or technical requirements, as they define the specific conditions or rules that must be met. This approach ensures that you verify and validate both functional and nonfunctional aspects of a feature.

In contrast, **scenario-oriented acceptance criteria** work best for user stories that involve user interactions or end-to-end workflows. This is because they illustrate the specific steps a user will take and are meant to ensure that the feature works as intended.

Best Practices for Writing Acceptance Criteria

Use Simple and Clear Language: Acceptance criteria should be easy to understand. Avoid technical jargon and write in plain language that is accessible to all stakeholders, including non-technical ones.

User-Centric: Frame your criteria from the user's perspective. What should they be able to achieve with completed functionality?

Be Specific and Measurable: Each criterion should be specific and measurable. Ambiguities should be minimized to avoid misinterpretations. For instance, instead of saying "The page should load quickly," specify "The page should load within 2 seconds."

Independent: Each criterion should be testable independently. This allows for incremental development and testing.

Define the Conditions of Satisfaction: Clearly state the conditions that must be met for the user story to be accepted. This can include functional requirements, performance criteria, and constraints.

Verifiable: Ensure each criterion can be demonstrably true or false through testing or validation.

Include Negative Scenarios: Consider edge cases and negative scenarios. Define how the system should behave in unexpected or erroneous situations.

Use the Given/When/Then Format: A popular format for writing acceptance criteria is the [Given/When/Then](#) template, borrowed from Behavior Driven Development (BDD). This format helps structure the criteria consistently and clearly:

- **Given:** The initial context or state of the system.
- **When:** The action or event that triggers functionality.
- **Then:** The expected outcome or result.

Examples of Acceptance Criteria

Example 1: User Login

User Story: As a user, I want to log in to the application so that I can access my account.

Acceptance Criteria:

- Given a registered user, when they enter a valid username and password, then they should be redirected to the dashboard.
- Given a registered user, when they enter an incorrect password, then they should see an error message “Invalid username or password.”
- Given an unregistered user, when they attempt to log in, then they should see an error message “User does not exist.”
- Given a registered user, when entering a valid username and password, then a session should be created.

Why are acceptance criteria important?

Acceptance criteria are crucial because they:

- Provide clear, testable requirements that help developers understand what needs to be built.
- Ensure that the product meets the expectations of the users and stakeholders.
- Facilitate communication and alignment among the Product Owner, developers, and stakeholders.
- Serve as the basis for creating test cases.

Who is responsible for writing acceptance criteria?

- Typically, the **Product Owner** is responsible for writing acceptance criteria, often with input from the developers, stakeholders, and sometimes even users. Collaboration ensures that criteria are comprehensive and aligned with user needs

and technical feasibility. The product owner typically takes the lead, but developers and other stakeholders can provide valuable input to ensure the criteria are clear, measurable, and achievable

How detailed should acceptance criteria be?

- Acceptance criteria should be detailed enough to be unambiguous but not so detailed that they stifle development flexibility. They should focus on the “what” (desired outcome) rather than the “how” (implementation details).

Can acceptance criteria change during the sprint?

- Ideally, acceptance criteria should be well-defined before the sprint begins. However, they may be refined during the sprint if new information or insights emerge. Any changes should be agreed upon by the Product Owner and the development team.

How many acceptance criteria should a user story have?

- There is no strict rule for the number of acceptance criteria a user story should have. However, they should be concise and comprehensive enough to cover all aspects of the user story. Too many criteria might indicate that the user story is too large and could be broken down further.

How do acceptance criteria relate to testing?

- **Acceptance criteria serve as the basis for creating test cases.** They define what needs to be tested and provide clear, testable conditions. Passing these tests indicates that the user story meets the defined requirements and is ready for acceptance.

Can acceptance criteria include non-functional requirements?

- Yes, acceptance criteria can and should include non-functional requirements when relevant. These might cover performance, security, usability, and other quality attributes that are critical to the functionality of the user story.