Engineer

Acceptance criteria

conditions a specific change must meet to be accepted

What it is

Acceptance criteria are the conditions a specific change must meet before it can be accepted. They describe what someone can observe under stated circumstances. “Send a booking confirmation” names a feature. “After an available appointment is booked, the confirmation shows the same appointment reference and start time as the booking record” gives a way to verify part of the feature. It leaves the team free to choose how the confirmation is generated.

The UK Government Service Manual describes acceptance criteria for a user story as an outcome checklist for confirming that the service has done its job. The story captures the user and the need. The criteria make the agreement concrete enough to discuss, implement and verify. A criterion can address an error path, an accessibility requirement or an operational constraint if it materially affects whether this change is acceptable. Different criteria may need different kinds of evidence.

Consider the relationship to three other artifacts. A specification can contain requirements that apply to many changes. A test case is a particular execution or example that produces evidence for one or more criteria. The Definition of Done in Scrum describes the quality measures that all work in an Increment must meet. Individual criteria address the behavior or limits of this change. For instance, a team might check keyboard reachability for every interactive change and also require that a particular appointment can be confirmed and revisited with the keyboard.

Use the concept in a decision

Put “What are acceptance criteria?” to work in a real decision. We can help relate the concept to the system you are building or buying. Turning what the business wants into something a developer can build, and saying out loud what is not worth building.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. A consistent identity across channels — 200x brand recognition, built from nothing

A consistent identity across channels Get in touch →

Why it matters

Good criteria make disagreements visible while a change is still cheap to adjust. A project sponsor may say that booking is successful when an appointment exists. A customer may expect the confirmation message to arrive, and support may need a reference when that message gets lost. Those are different outcomes. Deciding when to show success, what to do after an email provider fails and which reference is authoritative avoids discovering the conflict after release.

They also give verification a destination. A reviewer can inspect the booking page and a test operator can interrupt message delivery. Each knows what observation would count. Accessibility may need a keyboard walkthrough in addition to an automated assertion. Timing might need a measurement under a named load and a known test environment. A condition that only says the system must be “fast” leaves several incompatible choices unresolved.

Standards and regulations still apply without a line in the story. List a change-specific interpretation when the team would otherwise need to guess. If keyboard support is already a shared rule, describe how it applies to the unfamiliar date picker and use the existing standard as evidence. That keeps the story brief while protecting the behavior readers and operators need.

What it looks like in practice

Imagine a service that lets people reserve an appointment and receive a confirmation message. A first draft might say “the appointment can be booked and an email is sent”. Bring a service owner, engineer and tester together and walk through three concrete attempts. The following examples are illustrative. Their exact wording would come from the team operating that service.

- With an available appointment at 14:00 in the stated time zone, one confirmed booking is created and its reference, date and time appear on the confirmation page. Compare the values with the booking record.

- When the appointment is already taken by the time the request reaches the service, the user sees that the slot is unavailable and can return to available times. The competing request creates no additional booking for the occupied slot.
- Once the service has confirmed the booking, a simulated message delivery failure leaves that booking intact. The confirmation page still supplies the reference, and the failure is visible to the operator responsible for follow-up.

The examples state a starting situation, an action and a verifiable result. Gherkin expresses that structure with Given, When and Then. That format can improve readability and connect scenarios to automation. A short checklist works just as well for a simple change. In either form, specify which reference to compare and decide whose view supplies the evidence. A green status in a send queue cannot demonstrate that an email was received.

Run the first attempt through the actual screen. Test the competing update with the request ordering controlled. Simulate the message failure after the booking transaction has finished. Document the three observations in the work item so someone can reproduce the evidence. For a related example of specificity, see our accessibility statement, which describes keyboard checks and what was observed or still requires review. That is the level of evidence to ask for when a criterion depends on what a reader can actually do.

Where it goes wrong

Vague language moves the decision to the end. “The error is handled gracefully” is one example. Name the state, the information shown, the action still available and any effect that must be prevented. In the booking example those details tell a tester to verify both the visible recovery path and the booking record. A screenshot of the error message by itself leaves the second part unanswered.

Criteria can also prescribe implementation before it is justified. “Use the new queue” may be a legitimate architecture constraint, but it will not tell a user what happens when a confirmation is delayed. Keep the architecture decision where it can be explained and frame the criterion around observable behavior and evidence. If a specific protocol or storage location is compulsory for an external integration, cite that requirement and explain the boundary.

The opposite problem is a list so large it describes an entire release. Split the work by outcome and keep the important failure path with its corresponding behavior. Agree which constraints are shared and who verifies this particular change. When a prototype or test reveals a missed requirement, update the criterion with its reason and rerun the affected verification. Recording that change preserves the agreement that was used, so the next person can see both what passed and what remains open.

Further reading

- UK Government Service Manual on user stories and acceptance criteria
- Cucumber's Gherkin reference for scenarios and examples
- The Scrum Guide's Definition of Done
- W3C guidance on testing keyboard functionality

Glossary

Choose for your actual constraints

Bring the decision, the people affected and the options already considered. We can make the tradeoffs explicit and leave a documented recommendation.

See what it costs Get in touch →

Make the terminology practical

Ask how "What are acceptance criteria?" applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/acceptance-criteria/