# Writing Effective User Stories

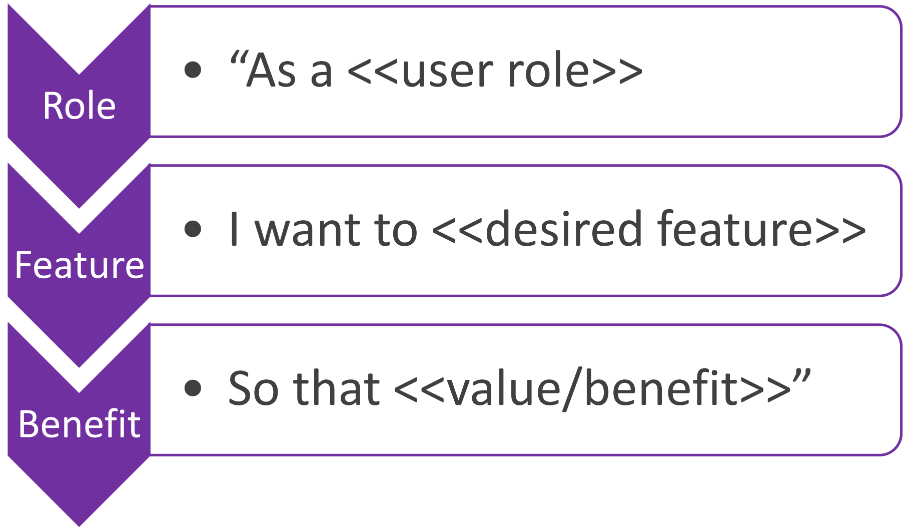
**Good User Stories have the following:**

- Goal oriented - deliver value to the customer.
- Contain only one viewpoint.
- Are written in active voice - i.e. " AS the end user, I am able to …" instead of "The button may be pushed by the end user."
- Describe how functionality *should* work, not how it *shouldn't*.
- Have a title that accurately reflects the content of the story.
- Have constraints/assumptions in the NOTES – i.e. "user must be in *sudo* to execute command, Users logged in as *ROOT* can perform action OR Command must be run from *user* directory".

# User Story Template

The User Story template captures the value of a functionality from the user's perspective.

- User Role – a user of the product
- Desired Feature – feature user needs
- Value/Benefit – why feature is important



- "As a <<user role>>
- I want to <<desired feature>>
- So that <<value/benefit>>"

**User Story Components:**
- The User Story "Sentence"
- Acceptance Criteria
- Assumptions
- Dependencies

# Writing Effective User Stories

An effective User Story follows the **INVEST** model

- **Independent –** avoids dependencies between stories
- **Negotiable** – agreement between the “what” and not the “how”
- **Valuable** – benefits to the customer are readily apparent
- **Estimable** – effort can be estimated
- **Sized Appropriately** – small enough to complete in a sprint
- **Testable** – defines done state and ensures quality

## Writing Acceptance Criteria

**Acceptance Criteria:**

- Include pass/fail test which describes what the story is to accomplish.
- Defines the boundaries for the story.
- Describes any follow-on actions during or after the user performs the activity, such as mouse-overs.
- Avoids keywords such as all, any, every, if appropriate, etc.
- Lists the individual roles, or group of roles, of those who are able to perform the activities in the story

**Actual Examples:**

- “End-to-end test results exported to log file”
- “upon SEBA install, complete status will be displayed.”

## *Example*: Acceptance Criteria

**User Story:** As a customer, I want to order and pay for the book via a secure web-based form, so that my credit card information is safe.

**Acceptance Criteria:**

1. All mandatory fields must be completed before a customer can submit a form.
2. Information from the form is stored in the customer orders database.
3. Payment can be made via Amex, Master Card, or Visa credit card.
4. The system shall accurately calculate and apply sales tax.
5. The system shall accurately calculate and apply shipping charges.
6. The customer shall be able to verify the accuracy of the order.
7. An acknowledgment email is sent to the customer submitting the form.
8. Protection against spam is working.

## Assumptions and Dependencies

**Assumptions**

- A condition that impacts the user story that the team believes is true
- Similar to assumptions in business requirements

*Actual examples of assumptions:*

- "All systems will report issues in the same way."
- "Provided user rights are in scope"
- "When requesting the transmission of message, Server communication must be present …"

**Dependencies**

- When the user story requires completion of another activity in order to either begin development or achieve functionality when deployed.

## User Stories: Example

**The User Story "Sentence"…**

- As a user, I want to be able to see available <u>Upgrades</u> when I am managing the service, so that I can effectively manage this service.

**Any relevant assumptions**

- Customer must be logged in to management GUI.
- Customer must be logged in as SUDO to clicking on 'check upgrade'.

**Acceptance criteria**

- AC1: <u>Upgrade Information</u> is displayed via popover window when user clicks on 'check upgrade'.
- AC2: User can click on 'continue' in popover, moves to 'upgrade screen'.
- AC3: User can click on 'cancel' in popover returns to previous page.

**Any dependencies??**

# Types of Backlog Items

**Non-User Stories** – *Backend functions/technical debt*

- Statements on infrastructure/technical needs that help deliver User Stories in the Product Backlog. Complete backend functions or address technical debt
    - *Example:* As a Developer, I want to configure our assigned development region, so that we can begin developing User Stories in our backlog.
    - *Example:* As a Tester, I want to prepare a test lab, so that we have representation of all device types & operating systems we need to execute tests on.

**Spike Stories** – *Analysis/research*

- Gather information on technical or functional questions. Do not produce shippable product
    - *Example:* As a Systems Analyst, I need to check the feasibility of storing all the parameters captured from different systems.

## What to Avoid:

**Test Only Stories**

- Avoid stories aimed at only testing tasks. Normal User Stories will include test tasks.

**Documentation Only Stories**

- User Stories should not be used as a document or record only.

## Key Points:

- User Stories are concise, written descriptions of a piece of functionality that will be valuable to a user or owner of software.
- User Story template (Role>Feature>Benefit) used to capture the value of a functionality from user’s perspective.
- Utilize INVEST model to create effective User Stories.
- User Stories contain: User story template, acceptance criteria, assumptions and dependencies.