**USER STORIES**

* a key component of agile software development.
* short, simple descriptions of a feature or functionality from the perspective of a user
* to capture requirements in an agile project and help the development team understand the needs and expectations of the users.

**CHARACTERISTICS**

1.User-centric

2.Simple

3.Independent

4.Negotiable

5.Testable

6.Iterative

7.Role-based

8.Traceable

**PATTERN**

* **Role-Feature-Benefit pattern**
* ***As a [ type of user], I want [ an action], so that [ some reason]***
* "As a [persona]": Who are we building this for? We’re not just after a job title, we’re after the persona of the person. Max. Our team should have a shared understanding of who Max is. We’ve hopefully interviewed plenty of Max’s. We understand how that person works, how they think and what they feel. We have empathy for Max.
* “Wants to”: Here we’re describing their intent — not the features they use. What is it they’re trying to achieve? This statement should be implementation free — if you’re describing any part of the UI and not what the user goal is you're missing the point.
* “So that”: how does their immediate desire to do something this fit into their bigger picture? What’s the overall benefit they’re trying to achieve? What is the big problem that needs solving?
* EX: As Max, I want to invite my friends, so we can enjoy this service together.
* EX: As a manager, I want to be able to understand my colleagues progress, so I can better report our success and failures.

**WRITING**

->**INVEST** Principle of User story

->INVEST principle which expresses the quality of the user story because in base

a good software product is completely dependent upon a good user story.

1.**I**ndependent – Not dependent on other.

2.**N**egotiable –Includes the important avoid contract.

3.**V**aluable –Provide value to customer.

4.**E**stimable –It should be estimated.

5.**S**mall –It should be simple and small not complex.

6.**T**estable –It should be evaluated by pre-written acceptance criteria.

A close-up of a list of stories

Description automatically generated

**Independent**

An independent user story is one that is:

1) conceptually separate from other user stories, and

2) not reliant on the completion of other user stories.

For example, suppose that a scrum team is considering working on two user stories — we’ll call them A and B — during the same sprint. Work on story B cannot start until story A is done. If story A does not reach completion until the end of the sprint (or is unfinished), the team has put itself in a position where the likelihood of unfinished work is considerably higher. By doing so, the team has incurred schedule risk.

**Negotiable**

When a team approaches a user story as negotiable, it means the story is an “invitation to a conversation.” As the team gets more familiar with the story context, additional ideas may emerge, the wording of the story may change, and details may be added.

**Valuable**

This one is rather self-explanatory. A completed user story has value when it meets an actual customer need.

**Estimatable**

An [estimatable user story](https://blog.logrocket.com/product-management/planning-poker-agile-estimation-scrum/) is one that is well enough understood by team members to be able to determine its relative size. In practice, “relative size” means that given a set of user stories, team members can determine which stories are about the same size (T-shirt sizes such as small, medium, and large often serve as a helpful relative sizing construct.)

**Small**

The S in INVEST stands for small. A small user story is one that can be completed during a short time window — for example, no more than a few business days.

**Testable**

A testable user story is one that can be verified to have met the story’s acceptance criteria (also known as “conditions of satisfaction”).

->**3 Cs** in User Stories

1)Card –Write stories on cards, prioritize, estimate, and schedule it accordingly.

2)Conversation –Conduct conversations, Specify the requirements and bring clarity.

3)Confirmation –Meet the acceptance criteria of the software.

**USER STORY EX FOR WEBSITE:**

User story examples for ***login***

1.As a user, I want an option to stay logged in, so that I don’t have to enter my credentials every time.

2.As a user, I want to be able to reset my password if I forget it, so that I can regain access to my account.

3.As a user, I want to see an error message if I enter incorrect login details,

so that I know when my login attempt has failed.

User story examples for ***registration***

1.As a user, I want to log in via my social media accounts, so that I can quickly access the platform without creating a new account.

2.As a new user, I want to choose my own username and password during registration, so that I can personalize my login credentials.

3.As a new user, I want to provide my basic information during registration, such as name and date of birth, so that I can personalize my profile.

**USER STORIES WITH ACCEPTANCE CRITERIA EXAMPLES**

Acceptance criteria and user stories are critical for successful project delivery for any software development project. They are essential for documenting expectations from a product or solution and how the test team can evaluate it for acceptance. Let’s look at the definition of acceptance criteria, their connection to user stories, and their role in project success.

* As a *restaurant customer*, I want to reserve a table online, so I can ensure I have a place to eat at my preferred time.

Acceptance criteria: The system should show available times for reservations. After reservation, the customer should receive a confirmation.

* As *a music app user*, I want to create playlists, so I can organize my favourite songs. Acceptance criteria: The app should provide the option to create a new playlist and add songs to it. Users should be able to name their playlists and view them later.

A list of product with text

Description automatically generated with medium confidence

**USER STORIES WITH REQUIREMENTS EXAMPLES**

Requirements typically refer to the documented expectations and specifications to meet a particular product need. They are a concise description of the feature or functionality that the software must provide.

* As a banking customer, I want to transfer funds between my accounts, so I can manage my money effectively.

Requirements: The system should allow customers to select two accounts, enter an amount, and execute a transfer. The system should update the account balances immediately.

* As a job seeker, I want to save job listings I’m interested in, so I can apply for them later.

Requirements: The job platform should allow users to save job listings to a personal list, which they can view later.

A screenshot of a white and black list

Description automatically generated

**USER STORY FOR SHOPPING APPLICATION**

A diagram of a shopper

Description automatically generated

* USER STORY: As a user, I want to log in via my social media accounts, so that I can quickly access the platform without creating a new account.

ACCEPTANCE CRITERIA:

* to choose my own username and password during registration
* can personalize my login credentials.
* want to provide my basic information during registration.
* USER STORY: As a user, I want an option to stay logged in, so that I don’t have to enter my credentials every time.

ACCEPTANCE CRITERIA:

* able to reset my password if I forget it
* can regain access to my account.
* to see an error message if entered incorrect login details
* USER STORY: As a shopper I want to view a list of products so I can select some to purchase.

ACCEPTANCE CRITERIA:

* See a thumbnail image for each product
* Click to view details for product
* Add to cart from detail page
* Search for a product
* View products by category
* View quantities and items in the cart
* USER STORY: As a shopper I want to review my cart so I can make adjustments prior to checkout

ACCEPTANCE CRITERIA:

* See a total cost before tax and shipping
* Remove items
* Adjust quantity of items
* Click to navigate to a product detail page
* Trigger checkout from any page, if there are items in the cart
* USER STORY: As a shopper I want to check out so I can get my products shipped to me

ACCEPTANCE CRITERIA:

* Enter a shipping address
* Enter a billing address
* Enter a credit card number
* Show total including tax and shipping before finalizing
* Show confirmation message after finalizing Verify payment via our payment processor
* View a list of open and completed orders
* USER STORY: As a shopper I want to review my orders so I can see what I've purchased in the past

ACCEPTANCE CRITERIA:

* See the status of the order
* Navigate to the details of the order
* Include a tracking number if the order is shipped but not delivered Contact customer service about an order from the details page
* USER STORY: As an administrator I want to modify the list of products so I can adjust our offerings over time

ACCEPTANCE CRITERIA:

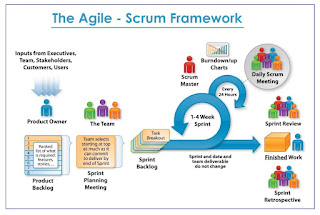
* Add or remove products
* Modify product images
* Select a category for the product

**SCRUM**

Scrum is a framework used primarily in agile software development for managing and completing complex projects. It emphasizes teamwork, collaboration, and iterative progress toward a well-defined goal. The term "Scrum" originates from rugby, where it refers to a method of restarting play after an infringement.

In software development, Scrum operates on the principles of transparency, inspection, and adaptation.

In scrum project move forward with series of iteration that are called Sprints. Each sprint size is typically two to four weeks long. It is based on inspect and adaptive cycle. Produce product incrementally and iteratively, thus reduce risk and enhance visibility.



**1 Product Owner**

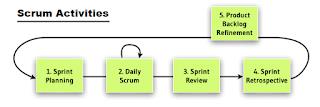
* Product Owner (PO) is client's representative, define features of product and decide release date and content
* Priorities features according to market value and be responsible for the profitability of product
* Accept or reject work items

**2 Scrum Master**

* Coach for scrum team , Enacting scrum values , Ensure team's productivity
* Build winning team, apply agile principles and make system effective.

**3 Team**

* 5-9 Members team (Developer , Tester) , Self-organizing, High performance team
* Build winning product, Work collaboratively and share responsibilities, Cross functional team.



Scrum Activities:  
1. Sprint Planning  
2. Daily Scrum  
3. Sprint Review  
4. Sprint Retrospective  
5. Product Backlog Refinement  
  
**1 Sprint Planning:**  
       Goal: Team to plan and agree on backlog items they can complete and confirm the tasks required to support acceptance

**2 Daily Scrum:**  
Goal: Plan for the day, Inspect and Adapt daily towards reaching the sprint goal.  
Description:

* Daily development Team standup for 15 minutes in circle and talk only on three points
* What I did since last daily scrum meeting?
* What I am planning to work on today?
* Impediments (Issue/blocker) if any?
* Scrum master protect the team and facilitate for being effective.
* This give an opportunity to team to inspect and adapt daily on the sprint goal.
* Who: Scrum Team, Scrum Master, When: Daily throughout the sprint , Duration: 15 minutes maximum

**3 Sprint Review:**  
Goal: Get feedback on product development. Inspect and adapt on the product feature.  
Description:

* During this meeting team demonstrate 100% completed work.
* Scrum master facilitate the environment.
* In case of new request, Product owner (PO) note and updates the product backlog as required.
* Product owner is final decision maker on acceptance.
* Duration: 2 hours for a 2 week sprint, Who: Scrum Team, Scrum Master, PO, Stakeholders, When: Last day of sprint

**4 Sprint Retrospective:**  
Goal: To inspect and adapt to become more effective and efficient on process, people, culture aspect.  
Description:

* Participation in the discussion to inspect and adapt as scrum team.
* Scrum master play vital role in sprint retrospective, Scrum master bring in the culture of openness, trust and respect as people discuss the improvement areas, facilitate and focus on improvement and changes that pointing fingers at others.
* This is platform to scrum master to help team resolve ineffectiveness in the systems
* Inspect and Adapt: Try everything that makes sense, reject things that didn’t work even after repeated trails. Shape your culture, process and practice.
* Duration: 2 hours for a 2 week sprint, Who: Scrum Team , When: Last day of sprint

**5 Product Backlog Refinement:**  
Goal: Keep product backlog items ready, uncertainty to certainty  
Description:

* Product owner provide clarity on each product backlog item (All uncertainty clarified into certainty )
* Product owner Update product backlog. 100% be present and involve all team members
* Team understand, carefully listen to need of product owner, understand the acceptance criteria. Help product owner to order the backlog.
* Duration: 1-3 hours depending on the team’s need. , Who: Scrum Team, Scrum master, PO, When: Continuous process, in between the sprints.

Scrum Artifacts:  
Below are Scrum Artifacts.  
1) Product Backlog,   2) Sprint Backlog, 3) Product Increment  
  
**1 Product Backlog**  
This is an ordered list of ideas for the product, which can come from the product owner, team members, or stakeholders. A description and estimate of effort complement each product backlog item.  
  
The product backlog is ordered to maximize the value delivered by the Scrum team. The development team’s work comes from the product backlog, and nowhere else. Every feature, enhancement, bug fix, documentation requirement, every bit of work the team does comes from a product backlog item.  
  
**2 Sprint Backlog**  
The sprint backlog is the list of refined product backlog items chosen for development in the current sprint, together with the team's plan for accomplishing the work. It reflects the team's forecast of what work can be completed. Once the sprint backlog is established, the development team begins work on the new product increment.  
  
**3 Product Increment**  
Every sprint produces a product increment, the most important Scrum artifact. A product Increment is the "goal line" for each sprint and, at the end of the sprint, it must:

* Be of high enough quality to be given to users
* Meet the Scrum team's current definition of done
* Be acceptable to the product owner

**PRODUCT OWNER**

a Product Owner is accountable for **maximizing the value** of the product resulting from the work of the Scrum Team. How this is done may vary widely across organizations, Scrum Teams and individuals.

As a member of the Scrum Team, the Product Owner provides clarity to the team about a product’s vision and goal. All work is derived and prioritized based on the **Product Goal** in order to deliver value to all stakeholders including those within their organization and all users both inside and out. Product Owners identify, measure and maximize value throughout the entire product's lifecycle.

### What does a Product Owner do?

The Product Owner is accountable for effective [Product Backlog](https://www.scrum.org/node/8119) management, which includes:

* Developing and explicitly communicating the Product Goal
* Creating and clearly communicating Product Backlog Items
* Ordering Product Backlog Items
* Ensuring that the Product Backlog is transparent, visible and understood

The Product Owner may do this work or delegate the responsibility to others on the Scrum Team. Regardless of who does the work, the Product Owner remains accountable for it being accomplished and for the **value delivered**.

Far beyond Product Backlog management, it is critical for the Product Owner to earn the **respect** of the entire organization in order to get the support they need for the decisions they make. This is key to a Product Owner’s success. These decisions need to be transparent in the Product Backlog, and through the [Increment](https://www.scrum.org/node/8121) of work shared at the [Sprint Review](https://www.scrum.org/node/8116).

Remember, the Product Owner is one person, not a committee. They also represent the needs of many stakeholders in the Product Backlog. If someone in the organization wants a change in the Product Backlog, they need to discuss this with the Product Owner and try to convince them. But at the end of the day, the Product Owner **makes the decision**. The Product Owner should also be getting feedback from customers on the product.

**PRODUCT BACKLOG**

## Examples of Product Backlog Items

Four main categories of items (called product backlog items) fit in the product backlog. Two are highly visible to customers — features and bugs. The other two, technical debt and research, are invisible to customers yet can't be ignored.

In an Agile organization, product backlog items are typically written as user stories — though they don’t always need to be. They can also be written as traditional requirements documents, or in a number of other ways.

When written as user stories, product backlog items often take the following form:

As a <stakeholder>, I want <action> so that <benefit>.

The agile product backlog in Scrum is a prioritized features list, containing short descriptions of all functionality desired in the product. When applying Scrum, it's not necessary to start a project with a lengthy, upfront effort to document all requirements. Typically, a Scrum team and its product owner begin by writing down everything they can think of for agile backlog prioritization. This agile product backlog is almost always more than enough for a first sprint. The Scrum product backlog is then allowed to grow and change as more is learned about the product and its customers.

A typical Scrum backlog comprises the following different types of items:

1. Features
2. Bugs
3. Technical work
4. Knowledge acquisition

By far, the predominant way for a Scrum team to express features on the agile product backlog is in the form of [user stories](https://www.mountaingoatsoftware.com/agile/user-stories), which are short, simple descriptions of the desired functionality told from perspective of the user. An example would be, "As a shopper, I can review the items in my shopping cart before checking out so that I can see what I've already selected."

Because there's really no difference between a bug and a new feature -- each describes something different that a user wants -- bugs are also put on the Scrum product backlog.

Technical work and knowledge acquisition activities also belong on the agile backlog. An example of technical work would be, "Upgrade all developers' workstations to Windows 7." An example of knowledge acquisition could be a Scrum backlog item about researching various JavaScript libraries and making a selection.  
  
The product owner shows up at the sprint planning meeting with the prioritized agile product backlog and describes the top items to the team. The team then determines which items they can complete during the coming sprint. The team then moves items from the product backlog to the sprint backlog. In doing so, they expand each Scrum product backlog item into one or more sprint backlog tasks so they can more effectively share work during the sprint.