10 PRODUCT STORY AUTHORING TIPS AND TRICKS

1. **Your story title should be 3-6 words and start with a verb**

   □ The story is about someone doing something; beware of nouns – e.g. "Submit button behavior"
   □ Don't write a novel in the title; that makes them hard to talk about

2. **Every story should have "Who? What? Why? How do I know I'm done?"**

   □ Who? – not "the users"; not a generalized user "IT administrator"; could be 1 or more personas
   □ Why? – value; What? – the thing you are building; When are you done? - varication
   □ Story templates are nice guidelines for consistency but aren't gospel.

3. **Stories describe an independent, testable, and valuable thing**

   □ Try to eliminate "dependency" thinking and move towards "incrementally growing product" thinking

4. **Writing less says more**

   □ Don't duplicate information. Use pictures, sketches, and white board snaps when you can
   □ Can anyone inside or outside the team read it and quickly understand what's going on?

5. **Group acceptance tests into categories**

   □ Easier to understand; provides outline for test plan
   □ If you need 50 AT's, the story is probably too big or you have written the test plan.
   □ Avoid repeating tests that apply to all stories or group them separately (e.g. browser support).

6. **Try to make your stories the same size (but don't get crazy)**

   □ Find the optimal size for your team (typically, 1/2 to 1/3 the cycle time) to avoid estimating costs.
   □ Still author smaller and larger stories when it makes sense.

7. **Be wary of too many spikes or technical stories**

   □ These are okay, but don't let them turn into a crutch for building testing/validation debt.
   □ Spikes must have acceptance tests that define "when we are done?" It's not a free for all.
   - Could be a time box – you spent 4 hours and reported what you discovered in that time.

8. Write stories last (after framing, personas, story mapping, journeys, etc.)

- □ Top-down design vs. bottom-up design
  - – Top-down tends to be much more user / product-based. - Bottom-up tends to be much more system-based / dev stack based.
- □ Manager says, “Go write ALL the stories for feature blah.” – no! Start with a map or titles.

9. Don’t write stories too soon

- □ Want to be at least 1 sprint ahead but no more than 3; too far ahead leads to rework
- □ Use story maps to show big picture, priorities, and progress.

10. Keep your story map up-to-date as you add and remove stories

- □ Product stories are difficult to visualize in backlog management tools. Pair a visual tool with a backlog management tool.
- □ Maps are also a tool for illustrating change/learning.