Lesson 4 Answers

Cleaning up

1 - Reflection questions

1.1 - Readability

We have emphasised the importance of readable scenarios. To whom is it important that they stay readable?

Readability matters most to the people on your team who can't read code. People like the business analyst, product owner, UX designer and and non-technical QA. If scenarios are readable, those people can use them as their "source of truth" as to how the system behaves

1.2 - The many roles of Gherkin

Gherkin serves several purposes at different stages in the software lifecycle. Can you name three of those stages and explain how Gherkin helps?

Specification, Development and Testing. Gherkin provides a place to tie all these together.

1.3 - Fresh Scenarios

What do we mean by keeping scenarios fresh and why is this important?

To maintain the team's trust in the scenarios, it's important that they always tell the truth about the current state of the system.

1.4 - Feature descriptions

What is the purpose of this section? What kind of information would you put in it?

Anything that's relevant to this feature. A user story is a start, but you can also include links to user research, external documents, issue tracker tickets etc. Try to make the feature file the central reference point for this feature: a living document.

1.5 - Domain language

Pick a scenario from your own project. Read the steps a few times. Does the language reflect the domain, or does it reflect the technical implementation?

*How could you improve the scenario to describe the **what** instead of the **how**?*

We can’t provide you with an answer for this one, but here’s a clue to test your own. Imagine delivering this same scenario through a completely different user interface. For example: if your system runs in a browser, imagine if you were building a mobile phone app instead that provides the same functionality. -Or perhaps a voice based UI. Would the scenario still hold true?